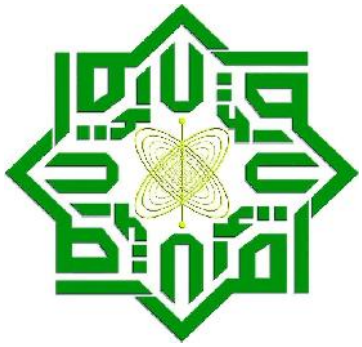
PENGEMBANGAN APLIKASI PENDETEKSI PLAGIARISME DOKUMEN DENGAN PENDEKATAN *K-GRAM* BERBASIS FRASA

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana
Teknik Pada Jurusan Teknik Informatika

oleh :

ADEK RAFLES
10851003270



FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI SULTAN SYARIF KASIM
PEKANBARU
RIAU

2013

PENGEMBANGAN APLIKASI PENDETEKSI PLAGIARISME DOKUMEN DENGAN PENDEKATAN *K-GRAM* BERBASIS FRASA

ADEK RAFLES

10851003270

Tanggal Sidang: 31 Januari 2013

Periode Wisuda : Februari 2013

Jurusan Teknik Informatika

Fakultas Sains dan Teknologi

Universitas Islam Negeri Sultan Syarif Kasim Riau

ABSTRAK

Plagiarisme merupakan sebuah tindakan penggunaan, mengutip sebagian atau keseluruhan isi karya tulisan orang lain tanpa mencantumkan sumber tulisan yang kemudian diakui sebagai miliknya sendiri. Ada banyak algoritma yang dapat mendeteksi plagiarisme dokumen teks salah satunya adalah pendekatan *k-gram*. Pada penelitian ini, pendekatan *k-gram* yang digunakan adalah token kata berbentuk *biword, triword* dan *quadword* secara bertalian. Konsep pendekatan ini, yaitu menemukan token kata *biword, triword* atau *quadword* yang sama dan terurut berdasarkan indeks di antara dua dokumen teks. Hipotesa awal, pendekatan token berbentuk *quadword* bekerja lebih baik dibanding token berbentuk *triword* dan *biword* dalam menemukan kutipan terpanjang yang sama di antara dua dokumen teks. Pengujian yang akan dilakukan meliputi aspek kuantitatif dan aspek kualitatif yang dihasilkan dari kombinasi token *biword, triword* dan *quadword* dengan batas terurut yang digunakan. Pada tahap pengujian, dokumen uji yang digunakan berjumlah tiga dokumen dengan kategori tingkat kemiripan rendah, sedang dan tinggi. Dari beberapa pengujian yang telah dilakukan, pendekatan ini dapat menemukan kutipan terpanjang yang sama di antara dua dokumen teks serta mengukur kemiripan dokumen teks. Selain itu, tahap pengujian mengasumsikan bahwa token *quadword* bekerja lebih baik dari token *triword* dan token *biword* dalam mendeteksi kutipan terpanjang yang sama di antara dua dokumen teks dengan batas minimal terurut >=4.

Kata kunci: *K-gram*, Plagiarisme Dokumen Teks, Token *Biword*, Token *Triword*, Token *Quadword*.

APPLICATION DEVELOPMENT DETECTION PLAGIRISM DOCUMENT BY K-GRAM APPROACH PHRASE BASED

ADEK RAFLES

10851003270

Final Exam Date: January 31th, 2013

Graduation Ceremony Period: February 2013

Information Engineering Department

Faculty of Sciences and Technology

State Islamic University of Sultan Syarif Kasim Riau

ABSTRACT

Plagiarism is a behavior use, cite partially or whole contain papers other author without publish source document then recognized as his self. There are many algorithm to detected plagiarism document text one of them is k-gram approach. In this research, k-gram approach used are word token shaped biword, triword and quadword consecutive. This approach concept, that is find the same word token biword, triword or quadword and ordered by index between two document text. Initial hypothesis, quadword token approach works better compared token triword and biword in found the same longest citation between two document text. The testing will be do involve quantitative aspects and qualitative aspect the resulting from combination token biword, triword and quadword with ordered threshold used. In the testing phase, document test used number three document test with the degree of similarity category low, medium and high. From some testing that has been done, this approach able found the same longest citation between two document text and measure of similarity document text. Furthermore, the phase test assume that token quadword works better from token triword and token biword in detecting the same longest citation between two document text with minimal ordered threshold >=4.

Key words: *K-gram, Plagiarism Document Text, Token Biword, Token Triword, Token Quadword.*

DAFTAR ISI

DAFTAR TABEL

Tabel **Halaman**

BAB I

PENDAHULUAN

1.1. Latar Belakang

Perkembangan informasi digital sangat memberikan pengaruh besar terhadap aktivitas kehidupan, baik dalam kehidupan sehari-hari, pendidikan, bisnis, perbankan, dan pemerintahan. Dengan adanya informasi digital, seseorang dapat mengetahui informasi baik berupa berita, promosi, peluang kerja, ilmu pengetahuan dengan cepat dan akurat melalui media internet. Selain itu, seseorang juga dapat berbagi karya tulis melalui media internet dengan maksud untuk berbagi ilmu dengan yang lain. Kadang kala, hal seperti ini sering sekali disalahgunakan bagi seseorang yang ingin mencari referensi atau bahan untuk menulis dengan melakukan *copy-paste* tanpa memahami isi dan mencantumkan sumber tulisan atau yang dikenal dengan tindakan plagiarisme.

Plagiarisme merupakan sebuah tindakan penggunaan atau mengutip sebagian isi karya tulisan orang lain tanpa mencantumkan sumber tulisan yang kemudian diakui sebagai miliknya sendiri. Plagiarisme mudah untuk dilakukan, hanya dengan menyalin, menempel, dan memodifikasi pada sebagian isi dokumen atau keseluruhan isi dokumen. Perbuatan seperti ini, merupakan sebuah perbuatan tidak baik yang dapat merugikan penulis asli, menghambat kreatifitas, menimbulkan kecenderungan sikap malas, tidak mau berfikir dan tidak mencerminkan sikap terpelajar bagi seorang siswa atau mahasiswa. Hal ini, semata-mata dilakukan untuk mempermudah dan mempercepat dalam menyelesaikan tugas seperti tugas praktikum, makalah, laporan kerja praktek bahkan skripsi atau tugas akhir.

Ada banyak algoritma yang dapat mendeteksi plagiarisme dokumen diantaranya algoritma *longest commonly consecutive word* (Sediyono, 2008), algoritma *winnowing* (Scheilmer, 2003), algoritma *rabin-karp* (Karp dan Rabin, 1978), algoritma *manber* (Manber, 1994), pendekatan kata *trigrams* (Lyon, 2001),

algoritma *longest common subsequence*, teknik *dot*, algoritma *boyer-moore* dan algoritma lainnya. Algoritma-algoritma ini, dapat diterapkan untuk mendeteksi bentuk plagiarisme seperti *verbatim copy* (menyalin kata perkata) atau *copy-paste* dan *pharafrase*.

Plagiarisme *verbatim copy*, dapat dideteksi dengan menggunakan teknik pemeriksaan kata bertalian yang sama dan terpanjang seperti algoritma *longest commonly consecutive word* (Sediyono, 2008). Algoritma ini, dapat menemukan kutipan terpanjang yang sama di antara dua dokumen teks serta lokasi dari kutipan tersebut. Kelemahan dari algoritma ini yaitu, membutuhkan waktu dalam melakukan proses pendeteksian plagiarisme dokumen teks. Selain itu, *verbatim copy* juga dapat dideteksi dengan menggunakan algoritma *winnowing* (Scheilmer, 2003) yang merupakan salah satu algoritma menggunakan pendekatan *k-gram* berbasis karakter dalam mendeteksi plagiarisme dokumen teks. Algoritma ini, efisien dalam mendeteksi plagiarisme dokumen teks dan menjamin bahwa kecocokan kalimat dengan panjang yang pasti dapat dideteksi.

Penelitian ini, bertujuan untuk menemukan kutipan terpanjang yang sama di antara dua dokumen teks dengan cara mencari irisan berurutan terpanjang dari token yang dihasilkan dua dokumen teks. Token ini, nantinya berbentuk *biword*, *triword* dan *quadword* secara bertalian. Hipotesa awal, pendekatan ini dapat menjaga makna atau arti kata dibandingkan algoritma *winnowing* yang membentuk sekumpulan karakter sehingga makna atau arti kata tidak dapat dikenali serta kehilangan sebagian kata-kata. Selain itu, hipotesa awal juga mengasumsikan pendekatan token berbentuk *quadword* bekerja lebih baik daripada token berbentuk *biword* dan *triword* dalam menemukan kutipan terpanjang serta mengukur kemiripan dokumen teks.

1.2. Rumusan Masalah

Berdasarkan penjelasan pada latar belakang diatas maka, dapat diuraikan rumusan masalah pada penelitian ini, yaitu:

1. Bagaimana membangun sebuah aplikasi yang dapat mendeteksi plagiarisme *verbatim copy* pada dokumen teks?

2. Bagaimana mengukur kemiripan dokumen teks dengan menggunakan *token* berbentuk *biword*, *triword* dan *quadword*?

1.3. Batasan Masalah

Batasan masalah dalam tugas akhir ini, yaitu:

1. Hanya membandingkan dua dokumen teks berbentuk *verbatim copy* atau *copy-paste*.
2. Tidak memperhatikan sinonim atau persamaan kata pada dokumen teks.
3. Tidak memperhatikan kalimat aktif dan kalimat pasif pada dokumen teks.
4. Tidak memperhatikan kata atau kalimat yang sama dari segi makna *(semantic)* pada dokumen teks.

1.4. Tujuan Penelitian

Tujuan yang ingin dicapai dalam pembuatan tugas akhir ini, yaitu:
Mendeteksi kutipan terpanjang yang sama di antara dua dokumen teks pada plagiarisme *verbatim copy* (menyalin kata perkata).

1.5. Sistematika Penulisan

Berikut merupakan rencana susunan sistematika penulisan laporan tugas akhir yang akan dibuat. Sistematika penulisan laporan tugas akhir ini meliputi:

Bab I Pendahuluan

Bab I ini merupakan bagian yang akan menjelaskan hal-hal seperti latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, dan sistematika penulisan laporan tugas akhir.

Bab II Landasan Teori

Pada bab ini dibahas mengenai pustaka atau literatur yang digunakan dalam pengerjaan skripsi. Teori-teori yang terdapat pada bab ini mencakup tentang plagiarisme, teknik *fingerprint matching*, pemrosesan dokumen,

jaccard coefficient, algoritma *winnowing*, algoritma *sieve of eratosthenes*, dan algoritma *longest commonly consecutive word*.

Bab III Metodologi Penelitian

Bab ini berisi tentang tahapan-tahapan yang dilakukan untuk menyelesaikan permasalahan pada tugas akhir ini seperti tahapan penelitian, studi pustaka, hipotesa, analisis aplikasi, perancangan aplikasi, implementasi aplikasi, pengujian aplikasi serta kesimpulan dan saran.

Bab IV Analisis dan Perancangan

Bab ini berisi tentang analisis dari penelitian yang dilakukan dalam tugas akhir ini seperti analisis pendekatan *k-gram*, analisis algoritma *sieve of erastothenes* serta gambaran umum aplikasi pendeteksi plagiarisme. Selain itu, pada bab ini juga menjelaskan tentang perancangan aplikasi.

Bab V Implementasi dan Pengujian

Bab ini berisi tentang penerapan aplikasi pendeteksian plagiarisme dokumen seperti tahapan implementasi. Selain itu, tahapan ini juga menjelaskan pengujian aplikasi.

Bab VI Penutup

Bab ini berisi kesimpulan dan saran mengenai hasil analisis, perancangan, hasil implementasi dan hasil pengujian yang telah dilakukan terhadap aplikasi pendeteksian plagiarisme dokumen.

BAB II

LANDASAN TEORI

2.1. Plagiarisme

2.1.1. Pengertian Plagiarisme

Plagiarisme berasal dari kata latin yaitu *plagiarius* yang berarti pencuri. Dari arti kata ini, dapat disimpulkan bahwa melakukan tindakan plagiarisme berarti mencuri hasil karya orang lain. Selain itu, plagiarisme juga dapat didefenisikan sebagai perbuatan mengambil hasil karangan orang lain dan mengakui sebagai hasil karangan sendiri atau mengutip karya tulisan seseorang tanpa mencatumkan sumber tulisan. Tindakan ini, dapat terjadi pada bidang apapun salah satunya pada bidang pendidikan. Hal ini, dikarenakan kurangya pemahaman seseorang baik siswa, mahasiswa atau lapisan elemen lainnya tentang plagiarisme dan pemahaman mengenai penulisan referensi.

Pada dunia pendidikan, plagiarisme dilakukan untuk mempermudah atau mempercepat dalam penyelesaian tugas baik bagi seorang siswa atau mahasiswa. Sebenarnya, setiap siswa atau mahasiswa telah mengetahui sanksi atau hukuman jika melakukan tindakan plagiarisme. Hukuman tersebut, dapat berupa dikeluarkan dari sekolah atau universitas tergantung kebijakan masing-masing institusi. Hal ini, tentu akan merugikan diri sendiri, orang lain, dan merusak nama baik institusi pendidikan terkait.

Bentuk tindakan plgiarisme ada beberapa macam dan tipe yang membedakannya. Ada beberapa perbedaan tindakan ini berdasarkan tipe, diantaranya (Martin, 1994):

1. *Word-for-word plagiarism*: menyalin secara langsung frasa, kalimat atau sebagian dari sumber tulisan dokumen teks tanpa mencantumkan sumber kutipan atau tulisan. Bentuk plagiarisme ini disebut juga *verbatim copy*.

2. *Paraphrasing plagiarism:* terjadi ketika kata-kata atau kalimat diganti dan ditulis ulang, akan tetapi tulisan tersebut masih bisa dikenali atau mirip dari karya tulis aslinya.
3. *Plagiarism of secondary sources:* terjadi ketika sumber tulisan yang asli dijadikan referensi atau dikutip, akan tetapi pada tulisan selanjutnya tidak melakukan kutipan langsung ke sumber tulisan aslinya.
4. *Plagiarism of the form of a source:* terjadi jika struktur pendapat dari sebuah sumber tulisan disalin secara kata demi kata atau ditulis ulang *(verbatim or rewritten).*
5. *Plagiarism of ideas:* menggunakan kembali sebuah gagasan atau ide (kecuali sebuah idea tau gagasan yang bersifat pengetahuan umum) dari sumber tulisan tanpa mengacu pada kata-kata atau bentuk sumber tulisan *(dependence on the words or form of source).*
6. *Plagiarism of authorship:* terjadi ketika mengakui hasil karangan orang lain sebagai tulisan sendiri dengan cara mencantumkan nama sendiri dan mengganti nama pengarang aslinya

2.1.2. Metode Pendeteksi Plagiarisme

Wang Tao (2008) mengatakan ada tiga metode atau pendekatan yang dapat dilakukan untuk mendeteksi plagiarisme dokumen teks, diantaranya adalah sebagai berikut:

1. *Grammar-based method*

   Metode ini fokus pada struktur tata bahasa dari dokumen dan menggunakan sebuah pendekatan *string-based matching* untuk menentukan kemiripan antara dokumen. Algoritma yang digunakan pada metode ini yaitu algoritma *longest common subsequence*, algoritma *winnowing* dan *hashbreaking*. Dengan menggunakan *grammar-based method* untuk mendeteksi plagiarisme *verbatim copy*, maka hasil yang didapatkan akan lebih baik dari pada menggunakan metode ini untuk mendeteksi dokumen teks yang memuat *sinonim* atau penulisan ulang *(rewritten).*

2. *Semantics-based method*

   Metode ini menggunakan model ruang vektor yang terdapat pada sistem temu kembali, statistik frekuensi kata di dalam sebuah dokumen digunakan untuk memperoleh fitur vektor dari dokumen, kemudian menggunakan *dot product*, *cosine*, dan sebagainya untuk mengukur fitur vektor dua dokumen. Fitur vektor ini merupakan kunci dari kemiripan dokumen. Metode ini, tidak senantiasa efektif untuk mendeteksi bagian dokumen yang telah diplagiasi karena metode ini sulit untuk menentukan letak atau posisi teks yang telah dijiplak.
3. *Grammar semantics hybrid method*

   Metode ini digunakan untuk mendeteksi bentuk plagiarisme *verbatim copy* dan *pharafrase*.

Algoritma *longest commonly consecutive word* sendiri termasuk pada *grammar-based method* karena algoritma ini digunakan untuk mendeteksi plagiarisme *verbatim copy* (menyalin kata perkata).

2.1.3. Kebutuhan Mendasar Algoritma Pendeteksi Plagiarisme

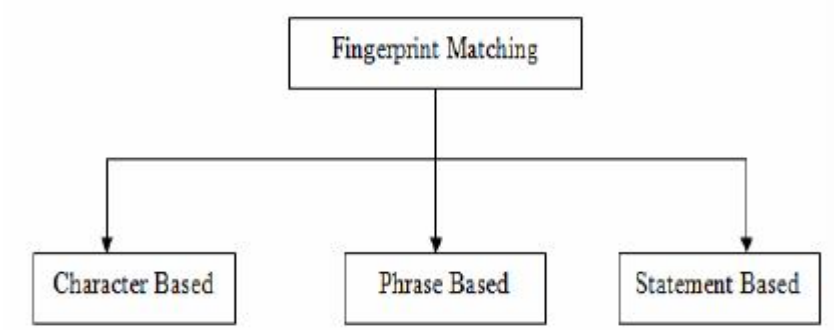
Untuk melakukan pendeteksian plagiarisme dokumen terdapat kebutuhan mendasar yang harus dipenuhi oleh suatu algoritma seperti (Scheilmer, 2003):

1. *Whitespace Insensitivity*, yang berarti dalam melakukan pencocokan terhadap dokumen teks seharusnya tidak terpengaruh oleh spasi, jenis huruf (kapital atau normal), tanda baca dan sebagainya.
2. *Noise Surpression*, yang berarti menghindari penemuan kecocokan dengan panjang kata yang terlalu kecil atau kurang relevan, misal: ‘the’. Panjang kata yang ditengarai merupakan penjiplakan harus cukup untuk membuktikan bahwa kata-kata tersebut telah dijiplak dan bukan merupakan kata yang umum digunakan.
3. *Position Independence*, yang berarti penemuan kecocokan atau kesamaan tidak harus bergantung pada posisi kata-kata. Meskipun berada pada posisi yang tidak sama, kecocokan atau kesamaan harus dapat ditemukan.

2.2. Teknik *Fingerprint Matching*

K-gram merupakan metode yang digunakan untuk membentuk *substring* sepanjang *k* karakter atau potongan sejumlah *k* karakter dari sebuah *string*. Biasanya yang dijadikan *substring* adalah kata. Metode ini, ditujukan untuk pembangkitan kata atau karakter dan digunakan untuk mengambil potongan-potongan karakter huruf sejumlah *k* dari sebuah kata secara kontinuitas atau berlanjut dibaca dari awal dokumen teks hingga akhir dari dokumen teks.

K-gram merupakan salah satu teknik *fingerprint* yang paling banyak digunakan. Teknik ini, membagi dokumen ke dalam *gram* sepanjang *k-gram* untuk menemukan *fingerprint* dokumen. Kemudian, *fingerprint* tersebut dibandingkan untuk mendeteksi kesamaan dokumen. Ada tiga pendekatan *fingerprint matching* berbasis *k-gram* (Chow Kok dan Naomie Salim, 2010):



Gambar 2.1. Teknik *Fingerprint Matching* (Chow Kok dan Naomie Salim, 2010)

Berikut ini penjelasan dari masing-masing pendekatan *fingerprint matching* berbasis *k-gram:*

1. *Character based*

   Merupakan teknik *fingerprint* yang menggunakan rangkaian karakter untuk mengetahui bentuk *fingerprint* dari sebuah dokumen teks. Pada tahun 1996, Heintze membagi teknik *fingerprint* ke dalam dua tipe yaitu *full* dan *selective*. Dalam penelitian ini, Heintze mengatakan bahwa nilai *k* yang efektif itu adalah 30-45 karakter.

2. *Phrase based*

   Lyon (2001) menggunakan *fingerprint* berbentuk frasa untuk mengukur kemiripan di antara dua dokumen teks. Langkah pertama pada pendekatan ini yaitu memotong setiap dokumen ke dalam *trigram* (tiga kata). Pendekatan ini bekerja lebih baik dan cepat daripada *character based fingerprint* karena mencocokan dengan kata lebih baik daripada mencocokan huruf.

3. *Statement based*

   Pendekatan ini diperkenalkan oleh Yerra dan Ng (2005) dengan menggunakan *fingerprint* dari tiap pernyataan (dari keseluruhan dokumen) dengan memilih sedikitnya 3 frekuensi 4-*gram* yang sama. Kelebihan dari metode ini adalah lebih efisien dari proses waktu dan ruang.

2.3. Pemrosesan Dokumen

Dalam ilmu sistem temu kembali informasi khususnya pada algoritma pendeteksi plagiarisme dokumen teks ada beberapa istilah yang tedapat dalam hal pemrosesan dokumen, diantaranya:

1. *Preprocessing* atau pembersihan teks merupakan tahapan yang dilakukan untuk mengubah data mentah menjadi data berkualitas yaitu data yang telah memenuhi persyaratan untuk diekseskusi pada sebuah algoritma. Bentuk pembersihan teks ini, dapat berupa menghilangkan spasi, tanda baca, simbol-simbol, mengubah huruf kapital menjadi huruf kecil dan menghilangkan karakter-karakter yang tidak relevan lainnya.
2. *Tokenizing* merupakan tahap pemotongan kalimat menjadi kata pada sistem temu kembali informasi. Pemotongan kata ini dapat berbentuk satu kata (*unigram* atau *uniword*), dua kata (*bigram* atau *biword*), tiga kata (*trigram* atau *triword*), empat kata (*quadgram* atau *quadword*) dan seterusnya.
3. Irisan *(intersection)* merupakan tahapan untuk menemukan kata bertalian yang sama di antara dua dokumen teks.

2.4. *Jaccard Coefficient*

Jaccard Coefficient merupakan persamaan yang digunakan untuk mengukur tingkat kemiripan antara dua dokumen teks. Berikut persamaan *jaccard coefficient*:

$$\text{Similaritas}(d_i,d_j) = \frac{|A(di) \cap B(dj)|}{|A(di) \cup B(dj)|} \quad \text{....................................(2.1)}$$

Keterangan:

$A(d_i)$: *fingerprint* dokumen teks 1

$B(d_j)$: *fingerprint* dokumen teks 2

2.5. Algoritma *Winnowing*

2.5.1. Pengenalan Algoritma

Algoritma berasal dari kata *algorism* merupakan nama seorang penulis buku arab terkenal yaitu Abu Jafar Muhammad Ibnu Musa Al-khuwarizmi (Al-Khuwarizmi dibaca orang barat menjadi *algorism*). Kata *algorism* kemudian berubah menjadi *algorithm* karena sering dikaitkan dengan ilmu *arithmathic* maka, akhiran -*sm* berubah menjadi -*thm*. Dalam bahasa Indonesia, kata *algoritm* diserap menjadi algoritma.

Algoritma (Rinaldi Munir, 2007) adalah urutan langkah-langkah dalam memecahkan atau menyelesaikan suatu permasalahan. Algoritma juga sering disebut sebagai jantung ilmu komputer atau informatika. Banyak cabang dari ilmu komputer yang mengacu pada algoritma, misalnya algoritma perutean *(routing)* pesan di dalam jaringan komputer, algoritma *Knuth-Morris-Pratt* untuk mencari pola di dalam teks dan algoritma *winnowing*.

2.5.2. Algoritma *Winnowing*

Algoritma *winnowing* (Scheilmer, 2003) merupakan urutan langkah-langkah untuk melakukan proses sidik jari dokumen *(document fingerprinting)*. Algoritma ini, merupakan salah satu algoritma pendeteksian plagiarisme berbasis

k-gram atau_*n-gram.* Algoritma ini, digunakan untuk pendeteksian plagiarisme dokumen teks dengan mengidentifikasi bagian-bagian terkecil yang mirip pada dokumen teks yang panjang. Algoritma ini, memiliki keunggulan dibandingkan algoritma dokumen *fingerprint* lainnya seperti algoritma *manber* dan algoritma *rabin-karp.* Hal ini disebabkan, algoritma *winnowing* dapat memberikan suatu hasil lebih informatif karena terdapat informasi posisi *fingerprint* dan memberikan jaminan terdeteksinya dokumen teks.

Penelitian mengenai algoritma *winnowing* sudah ada yang melakukan salah satunya dilakukan oleh Putu Yuwono (2010). Penelitian ini, menghasilkan kombinasi terbaik dalam menentukan nilai variabel-variabel yang terdapat pada langkah-langkah penerapan algoritma *winnowing* yaitu n = 30, b = 3, w = 30 serta *threshold* similaritas yang dianggap plagiat >=50%.

2.5.2. Langkah-Langkah Algoritma *Winnowing*

Ada langkah-langkah yang dilakukan dalam menerapkan algoritma *winnowing* adalah sebagai berikut:

1. *Preprocessing* atau pembersihan teks merupakan tahap melakukan proses *white insentivity*, dengan mengubah huruf kapital menjadi huruf kecil, menghilangkan tanda baca atau simbol-simbol, spasi dan karakter-karakter yang tidak relevan lainnya. Misalnya:
   `“Saya mahasiswa Teknik Informatika UIN SUSKA RIAU”`
   Sehingga kalimat diatas menjadi:
   `sayamahasiswateknikinformatikauinsuskariau`

2. Pembentukan *substring* dari dokumen teks menggunakan metode *k-gram.*
   K-gram merupakan metode yang digunakan untuk membentuk *substring* sepanjang *k* karakter atau potongan sejumlah *k* karakter dari sebuah *string*.
   Berikut ini hasil rangkaian *k-gram* dengan nilai *k* = 9:

sayamahas	ayamahasi	yamahasis	amahasisw	mahasiswa
ahasiswat	hasiswate	asiswatek	siswatekn	iswatekni
swateknik	watekniki	ateknikin	teknikinf	eknikinfo
knikinfor	nikinform	ikinforma	kinformat	informati

```
nformatik  formatika  ormatikau  rmatikaui  matikauin
atikauins  tikauinsu  ikauinsus  kauinsusk  auinsuska
uinsuskar  insuskari  nsuskaria  suskariau
```

3. Melakukan perhitungan nilai *hash* dari setiap *gram* yang terbentuk. Ada persamaan dari metode *hash*, yaitu:

   $H_{(c1….ck)} = c_1 * b^{(k-1)} + c_2 * b^{(k-2)} + \cdots + c_{(k-1)} * b^k + c_k$ …….(2.2)

   Keterangan:

   c : nilai *ascii* karakter (desimal)

   b : basis (bilangan prima)

   k : banyak karakter (indeks karakter)

   Untuk mendapatkan nilai hash dari metode *k-gram* selanjutnya digunakan persamaan *rolling hash* dibawah ini:

   $H_{(c2….ck+1)} = (H_{(c1….ck)} - c_1 * b^{(k-1)}) * b + c_{(k+1)}$…….(2.3)

   Rolling hash berfungsi untuk mempercepat komputasi nilai hash dari rangkaian *gram* selanjutnya yang telah terbentuk. Nilai *hash* yang baru dapat dengan cepat dihitung dari nilai *hash* yang lama dengan cara menghilangkan nilai lama dari kelompok *hash* dan menambahkan nilai baru ke dalam kelompok tersebut.

   Berikut ini hasil perhitungan dari nilai *hash* dimana nilai $b = 3$ dan $k = 9$:

1091224 1010232 1121560 983156 1040314 975611 1017683
1006124 1109231 1064253 1126151 1115013 1002872 1099467
1015284 1057983 1067977 1038898 1050095 1044309 1066319
1033924 1094223 1097961 1050131 1005061 1106049 1035034
1038494 1009498 1119357 1055265 1099177 1132518

Ada beberapa contoh cara perhitungan nilai-nilai *hash* diatas:

$H_{(sayamahas)}$ = ascii(s) * $3^{(8)}$ + ascii(a) * $3^{(7)}$ + ascii(y) * $3^{(6)}$ + ascii(a) * $3^{(5)}$ + ascii(m) * $3^{(4)}$ + ascii(a) * $3^{(3)}$ + ascii(h) * $3^{(2)}$ + ascii(a) * $3^{(1)}$ + ascii(s) * $3^{(0)}$

= 115 * 6561 + 97 * 2187 + 121 * 729 + 97 * 243 + 109 * 81 + 97 * 27 + 104 * 9 + 97 * 3 + 115 * 1

= 754515 + 212139 + 88209 + 23571 + 8829 + 2619 + 936 + 291 + 115

= 1091224

$H_{(ayamahasi)}$ = (1091224 - ascii(s) * $3^{(8)}$) * 3 + ascii(i) * $3^{(0)}$

= (1091224 – 115 * 6561) * 3 + 105 * 1

= (1091224 – 754515) * 3 + 105 * 1

= (336709 * 3) + 105

= 1010127 + 105

= 1010232

$H_{(yamahasis)}$ = (1010232 - ascii(a) * $3^{(8)}$) * 3 + asscii(s) * $3^{(0)}$

= (1010232 – 97 * 6561) * 3 + 115 * 1

= (1010232 – 636417) * 3 + 115 * 1

= (373815 * 3) + 115

= 1121445 + 115

= 1121560

4. Membentuk beberapa *window* dengan jumlah tertentu menggunakan konsep metode *k-gram*. Ada beberapa *window* yang dibentuk dengan menggunakan nilai *w* = 4:

[1091224, 1010232, 1121560, **983156**]

[1010232, 1121560, 983156, 1040314]

[1121560, 983156, 1040314, 975611]

[983156, 1040314, 975611, 1017683]

[1040314, **975611**, 1017683, 1006124]

[975611, 1017683, 1006124, 1109231]

[1017683, **1006124**, 1109231, 1064253]

[1006124, 1109231, 1064253, 1126151]

[1109231, **1064253**, 1126151, 1115013]

[1064253, 1126151, 1115013, **1002872**]

[1126151, 1115013, 1002872, 1099467]
[1115013, 1002872, 1099467, 1015284]
[1002872, 1099467, 1015284, 1057983]
[1099467, **1015284**, 1057983, 1067977]
[1015284, 1057983, 1067977, 1038898]
[1057983, 1067977, **1038898**, 1050095]
[1067977, 1038898, 1050095, 1044309]
[1038898, 1050095, 1044309, 1066319]
[1050095, 1044309, 1066319, **1033924**]
[1044309, 1066319, 1033924, 1094223]
[1066319, 1033924, 1094223, 1097961]
[1033924, 1094223, 1097961, 1050131]
[1094223, 1097961, 1050131, **1005061**]
[1097961, 1050131, 1005061, 1106049]
[1050131, 1005061, 1106049, 1035034]
[1005061, 1106049, 1035034, 1038494]
[1106049, 1035034, 1038494, **1009498**]
[1035034, 1038494, 1009498, 1119357]
[1038494, 1009498, 1119357, 1055265]
[1009498, 1119357, 1055265, 1099177]
[1119357, **1055265**, 1099177, 1132518]

5. Pilih nilai *hash* minimum dengan posisi paling kanan dari setiap *window* yang telah terbentuk untuk dijadikan *fingerprint* (sidik jari atau penanda) jika nilai tersebut berada pada posisi sama yaitu paling kanan maka kedua nilai *hash* tersebut dijadikan sebagai *fingerprint*. Berikut ini nilai *fingerprint* terkecil yang dihasilkan berdasarkan indeks:

[983156, 3] [975611, 5] [1006124, 7] [1064253, 9] [1002872, 12] [1015284, 14] [1038898, 17] [1033924, 21] [1005061, 25] [1009498, 29] [1055265, 31]

Sehingga *gram* yang digunakan berdasarkan *fingerprint* terkecil yaitu:

```
amahasisw  ahasiswat  asiswatek  iswatekni  ateknikin
eknikinfo  ikinforma  formatika  atikauins  auinsuska
insuskari
```

6. Untuk menghitung kemiripan antar dokumen digunakan persamaan *jaccard coefficient*.

   Similaritas(d_i,d_j) = $\frac{|W(di) \cap W(dj)|}{|W(di) \cup W(dj)|}$

   Keterangan:

   W(d_i) : *fingerprint* terkecil dokumen teks 1

   W(d_j) : *fingerprint* terkecil dokumen teks 2

   Misalkan, nilai *fingerpritnt* terkecil dari dua dokumen teks:

   D1 = [983156] [975611] [1006124] [1064253] [1002872] [1015284] [1038898] [1033924] [975611] [1038898]

   D2 = [973156] [935611] [1006124] [1005061] [1064253] [1002872] [1015284] [1038898] [1033924] [1005061]

   Similaritas(d_i,d_j) = $\frac{|W(di) \cap W(dj)|}{|W(di) \cup W(dj)|}$

   $= \frac{|[1006124][1064253][1002872][1015284][1038898][1033924]|}{11} = \frac{6}{11} = 54\%$

2.6. Algoritma *Sieve of Erastothenes*

Eratosthenes (276-194 SM) adalah seorang petugas perpustakaan ketiga dari perpustakaan terkenal di Alexandria dan juga merupakan seorang sarjana yang sangat hebat. *Erastothenes* dikenang dengan pengukurannya terhadap keliling bumi dengan memperkirakan jarak antara bumi dengan matahari dan bulan. Selain itu, ia juga menemukan sebuah algoritma untuk mencari bilangan prima yang dikenal dengan algoritma *Sieve Of Erastothenes* (Himawan, 2010).

Algoritma *Sieve Of Erastothenes* merupakan sebuah algoritma klasik untuk menemukan seluruh bilangan prima sampai ke sebuah N yang ditentukan. Selain itu, algoritma ini dapat juga digunakan untuk mempercepat solusi pada

permasalahan yang ada dengan melakukan sedikit modifikasi pada algoritma tersebut. Algoritma ini, nantinya penulis gunakan untuk mempercepat dalam mencari pasangan dari kumpulan kata bertalian yang telah terbentuk. Ada langkah-langkah algoritma *Sieve Of Erastothenes* dalam menemukan bilangan prima, yaitu:

1. Buat daftar bilangan mulai dari 2 sampai ke N.
2. Ambil bilangan terkecil dari daftar yang merupakan bilangan prima dan simpan dalam daftar baru.
3. *Eliminasi* bilangan dari kelipatan bilangan prima yang telah ditemukan.
4. Ulangi langkah 2 dan 3 sampai tidak ada bilangan yang tersisa dalam daftar.

Berikut contoh penerapan algoritma *Sieve Of Erastothenes* dalam menemukan bilangan prima. Misalkan N bilangan prima adalah sebagai berikut:

2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

Angka 2 merupakan bilangan prima maka *eliminasi* bilangan kelipatan 2 sehingga daftar bilangan menjadi:

2 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ 9 ~~10~~ 11 ~~12~~ 13 ~~14~~ 15 ~~16~~ 17 ~~18~~ 19

Angka selanjutnya 3 merupakan bilangan prima maka *eliminasi* bilangan kelipatan 3 sehingga daftar bilangan menjadi:

2 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ ~~9~~ ~~10~~ 11 ~~12~~ 13 ~~14 15~~ ~~16~~ 17 ~~18~~ 19

Ulangi langkah diatas sampai tidak ada bilangan yang tersisa dalam daftar sehingga diperoleh bilangan prima 2 3 5 7 11 13 17 19

2.7. Algoritma *Longest Commonly Consecutive Word*

2.7.1. Algoritma *Longest Commonly Consecutive Word*

Lokasi atau letak merupakan hal yang paling penting dalam pendeteksian plagiarisme dokumen teks. Hal ini, dibutuhkan untuk memberikan informasi posisi dari isi dokumen sebagai jaminan bahwa dokumen telah dijiplak. Selain itu, letak juga digunakan untuk menghitung kemiripan antar dokumen. Algoritma ini, menggunakan paragraf sebagai sebuah unit pembanding dan membagi paragraf tersebut ke dalam kumpulan kata-kata secara bertalian membentuk segitiga. Posisi

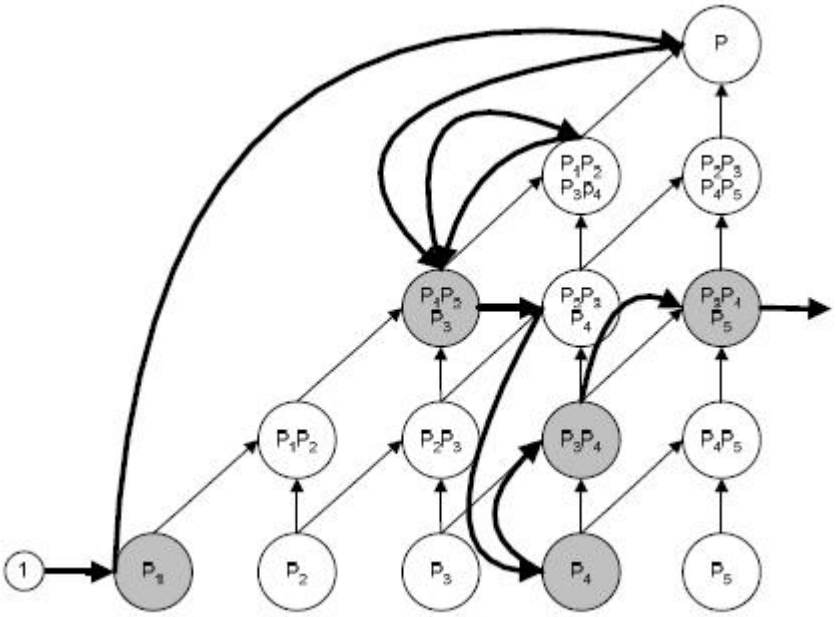
dari kata-kata pada paragraf direkam, kemudian kata-kata tersebut digunakan sebagai pembanding sehingga kutipan yang panjang dapat diidentifikasi dan posisi kemiripan teks pada dokumen dapat diperoleh menggunakan algoritma *longest commonly consecutive word* (Sediyono, 2008).

2.7.2. Langkah-Langkah Algoritma *Longest Commonly Consecutive Word*

Teks pada dokumen sumber dibagi ke dalam paragraf. Kemudian setiap paragraf diberi tanda dokumen dan identifikasi paragraf, *DocId:ParaId* sebagai sebuah identifikasi lokasi dari paragraf. Kemudian, teks di setiap paragraf dipenggal menjadi kata-kata yang bertalian. Ukuran yang paling kecil dari kata bertalian yaitu satu kata. Dari semua kata-kata yang bertalian ini, dibentuk sebuah segitiga untuk setiap paragrafnya.

Sebagai contoh, untuk setiap paragraf yang memiliki *m* kata, terdapat $P=\{P_1, P_2, P_3, P_4,...., P_m\}$ sebagai sebuah *base members* pada *level* terbawah (*level* satu) dari segitiga. Untuk level dua, setiap anggota *level* satu digabung dengan *level* satu lainnya seperti P_1P_2, P_2P_3, $P_3P_4....$, $P_{m-1}P_m$. Kemudian untuk *level* 3 anggotanya adalah P_1P_2 P_3, P_2P_3 P_4, P_3P_4 $P_5....$, $P_{m-2}P_{m-1}P_m$. Setelah itu, lakukan langkah seperti yang telah dijelaskan sebelumnnya sampai setiap kata terhubung satu sama lain.

Demikian juga dengan dokumen yang akan diamati, teks pada dokumen dibagi ke dalam paragraf dan kemudian untuk setiap paragraf dibangun sebuah segitiga kata bertalian seperti yang dilakukan pada dokumen sumber. Untuk menemukan *commonly consecutive words (CCWs),* titik-titik pada segitiga paragraf yang diamati dibandingkan dengan titik-titk paragraf sumber. Dengan mengetahui posisi dari titik pada segitiga dan juga *DocId:ParaId,* lokasi dari *CCW* dapat ditentukan dan kutipan terpanjang dapat ditemukan dari *CCW* yang terpanjang. Untuk menghindari pengecekan titik pertitik, dilakukan pendekatan dengan cara *diagonal search* dan *vertical search*. *Diagonal search* dilakukan untuk memulai titik sedangkan *vertical search* dilakukan apabila *diagonal search* menemukan *CCW* kecuali jika *CCW* adalah *base node.*



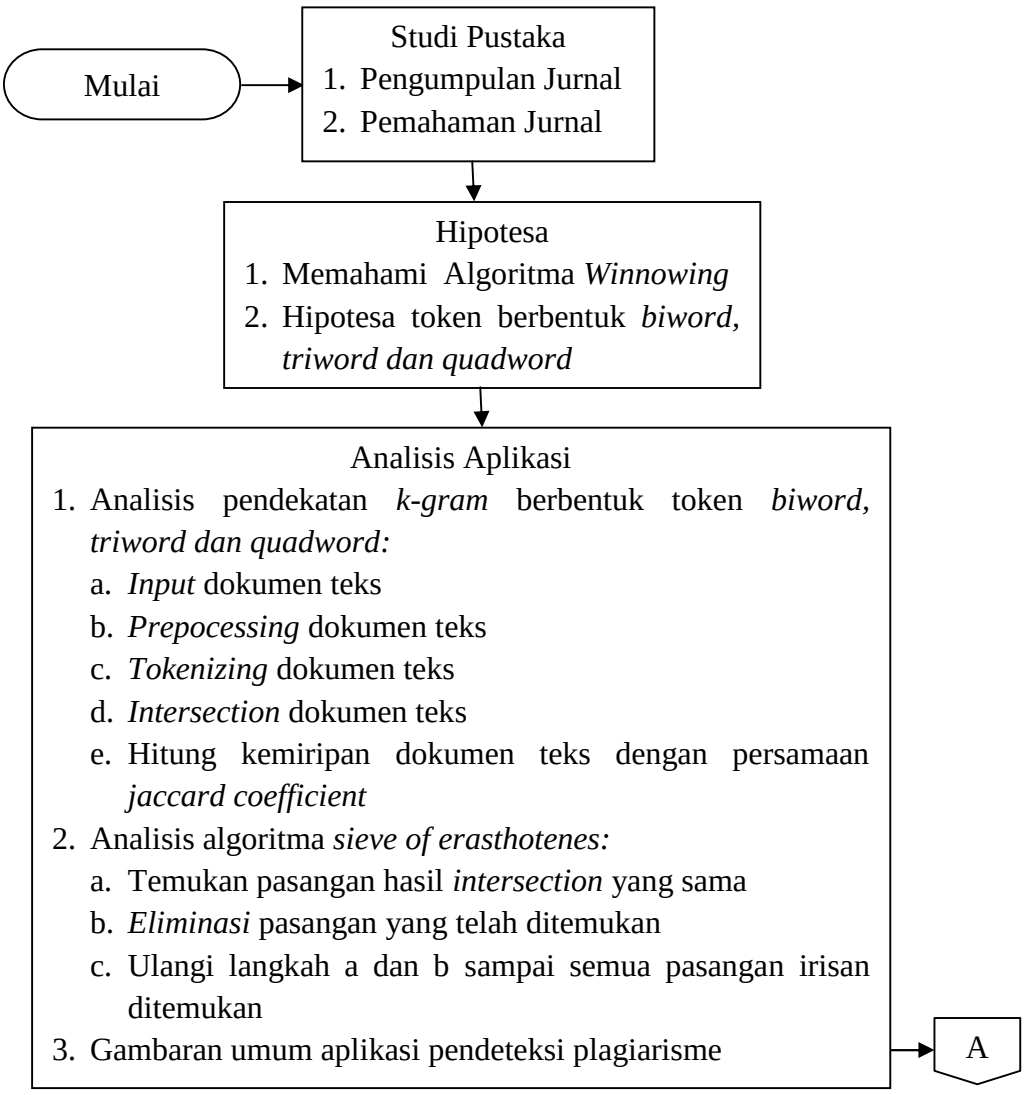
Gambar 2.2 Konsep Algoritma *LCCW* (Sediyono, 2008)

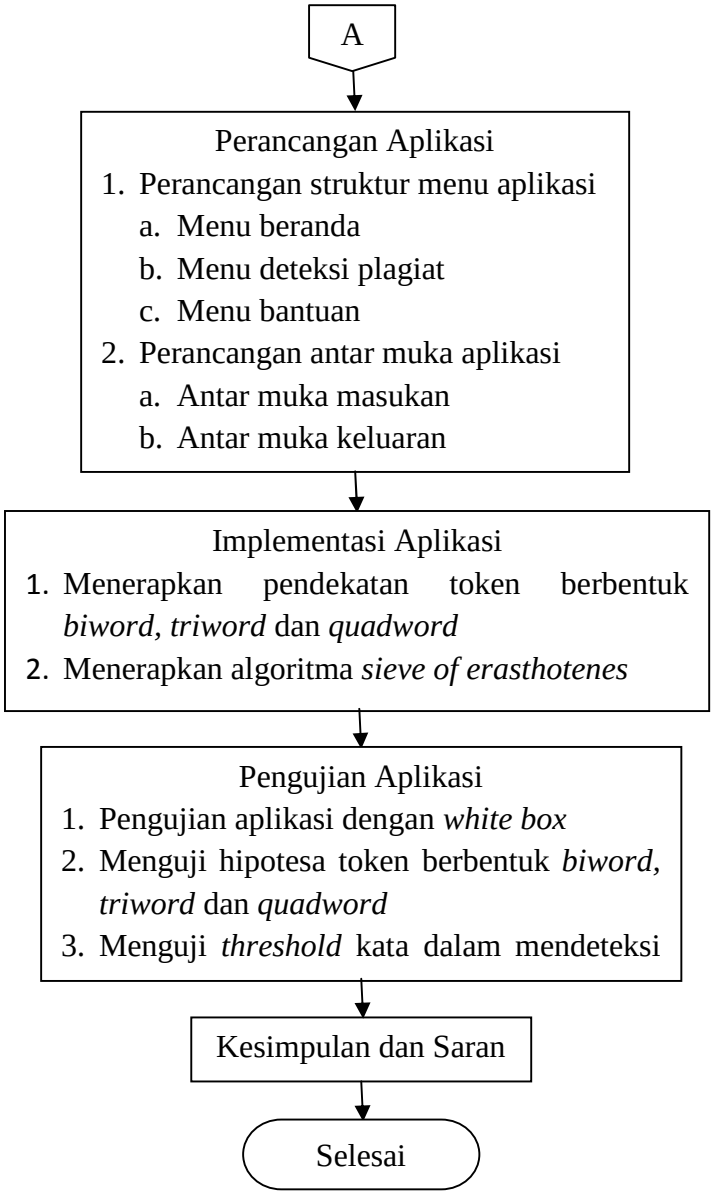
BAB III

METODOLOGI PENELITIAN

3.1. Tahapan Penelitian

Metodologi penelitian bertujuan untuk menggambarkan seluruh kegiatan yang dilaksanakan selama penelitian berlangsung. Ada beberapa tahapan yang akan dilakukan untuk menyelesaikan kasus pada penelitian tugas akhir ini yang meliputi: studi pustaka, hipotesa, analisis aplikasi, perancangan aplikasi, implementasi aplikasi, pengujian aplikasi, serta kesimpulan dan saran.





Gambar 3.1 Tahapan Metodologi Penelitian

3.2. Studi Pustaka

Studi pustaka merupakan metode yang bertujuan untuk memperoleh informasi-informasi atau data-data mengenai topik dan permasalahan pada penelitian ini. Ada dua metode yang dilakukan penulis dalam memperoleh informasi atau data pada penelitian ini:

1. Pengumpulan Jurnal

Metode ini dilakukan dengan cara mengumpulkan data atau informasi-informasi mengenai topik pada tugas akhir ini berupa jurnal-jurnal atau tulisan penelitian tentang algoritma pendeteksian plagiarisme atau artikel-artikel yang membahas kasus yang sama dengan kasus dalam tugas akhir ini.

2. Pemahaman Jurnal

   Metode ini dilakukan dengan cara mempelajari dan memahami jurnal tentang algoritma pendeteksian plagiarisme.

3.3. Hipotesa

Metode ini dilakukan dengan memahami algoritma *winnowing* sehingga diperoleh suatu hipotesa berupa pendekatan token berbentuk *biword, triword* dan *quadword.* Hipotesa awal, pendekatan token berbentuk *quadword* bekerja lebih baik dibanding token berbentuk *triword* dan *biword* dalam menemukan kutipan terpanjang yang sama di antara dua dokumen teks dengan mengambil token terurut minimal membentuk lima kata serta mengukur kemiripan dokumen teks menggunakan persamaan *jaccard coefficeient*.

3.4. Analisis Aplikasi

Analisis merupakan sebuah metode khusus yang digunakan untuk menganalisis masalah yang ada pada penelitian. Ada beberapa analisis aplikasi yang dilakukan, diantaranya:

1. Analisis langkah-langkah pendekatan *k-gram* berbentuk *biword, triword* dan *quadword* dalam mendeteksi plagiarisme dokumen teks. Ada langkah-langkahnya sebagai berikut:
   a. *Input* dokumen teks A dan dokumen teks B yang akan dideteksi plagiat.
   b. Lakukan *preprocessing* atau pembersihan teks pada dokumen teks A dan dokumen teks B seperti mengubah huruf kapital ke huruf kecil, menghilangkan spasi, tanda baca serta karakter-karakter yang tidak relevan lainnya.

c. Lakukan *tokenizing* berbentuk *biword, triword,* dan *quadword* pada dokumen teks A dan dokumen B yang telah dilakukan pembersihan teks.
d. Lakukan *intersection* antara dokumen teks A dan dokumen B serta dokumen teks B dan dokumen A sehingga diperoleh kata bertalian yang sama diantara dua dokumen teks.
e. Hitung kemiripan dokumen teks dengan menggunakan persamaan *jaccard coefficient.*

2. Analisis langkah-langkah algoritma *sieve of erasthotenes* dalam mempercepat pencarian pasangan hasil irisan yang sama. Ada langkah-langkahnya sebagai berikut:
   a. Temukan pasangan hasil irisan yang sama antara dokumen A dan dokumen B dengan dokumen B dan dokumen A kemudian simpan dalam daftar baru.
   b. *Eliminasi* pasangan irisan yang telah ditemukan dari hasil irisan antara dokumen A dan dokumen B serta dokumen B dan dokumen A untuk mempercepat dalam mencari pasangan irisan.
   c. Lakukan langkah a dan b sampai semua pasangan irisan ditemukan.
3. Gambaran umum aplikasi pendeteksi plagiarisme yaitu menjelaskan alasan yang melatarbelakangi dalam pembuatan aplikasi ini serta menganalisis kebutuhan dalam membangun aplikasi pendeteksi plagiarisme.

3.5. Perancangan Aplikasi

Perancangan aplikasi merupakan metode yang digunakan untuk merancang hal-hal yang telah dilakukan pada analisis dengan tujuan untuk memberikan gambaran terhadap aplikasi yang akan dibangun pada penelitian ini, diantaranya:

1. Rancangan tampilan struktur menu aplikasi dalam mendeteksi plagiarisme dokumen teks. Ada menu yang terdapat pada aplikasi ini, yaitu:
   a. Menu beranda, berisi tentang penjelasan aplikasi pendeteksi plagiarisme yang dibangun.

b. Menu deteksi plagiat, berisi antar muka aplikasi untuk mendeteksi plagiarisme dokumen teks.
c. Menu bantuan, berisi tentang cara menggunakan aplikasi pendeteksi plagiarisme yang dibangun.

2. Rancangan tampilan antar muka (*user interface*) aplikasi dalam mendeteksi plagiarisme dokumen teks. Ada antar muka yang terdapat pada aplikasi ini, yaitu:
   a. Antar muka masukan merupakan antarmuka untuk mendeteksi plagiarisme dokumen teks.
   b. Antar muka keluaran merupakan antarmuka untuk menampilkan kutipan yang sama di antara dua dokumen teks.

3.6. Implementasi Aplikasi

Implementasi aplikasi dilakukan setelah analisis dan perancangan rancang bangun aplikasi selesai dilakukan. Metode ini, akan menjelaskan tentang penerapan pendekatan token berbentuk *biword*, *triword* dan *quadword* dalam mendeteksi plagiarisme dokumen teks dan algoritma *sieve of erasthotenes* dalam mempercepat mencari pasangan irisan yang sama di antara dua dokumen teks serta jalannya rancang bangun yang telah dianalisis dan dirancang. Aplikasi yang telah dirancang dan dianalisis selanjutnya diimplementasikan dan dilakukan pengujian untuk mengetahui tingkat keberhasilan aplikasi yang telah dibangun. Implementasi pengembangan aplikasi ini akan dikembangkan pada spesifikasi *hardware* dan *software* sebagai berikut:

1. Perangkat keras

   Processor : *Intel(R) Core(TM) i3 CPU M330 @ 2.13GHz*
   Memori (RAM) : 2,00 GB
   Harddisk : 320 GB

2. Perangkat Lunak

   Sistem operasi : *Windows 7 Ultimate 32-bit OS*
   Bahasa pemrograman : *Hypertext Preprocessor (PHP)*
   Tool : *Notepad++*

3.7. Pengujian Aplikasi

Metode ini dilakukan untuk mengetahui tingkat keberhasilan aplikasi yang telah dibangun dengan melakukan ujicoba atau pengujian yang bertujuan untuk mengoptimalkan kinerja aplikasi. Ada beberapa pengujian yang akan dilakukan diantaranya:

1. Menguji aplikasi dengan *white box* bertujuan untuk mengetahui apakah algoritma yang telah diterapkan sudah benar dan sesuai.
2. Menguji hipotesa token berbentuk *biword, triword* dan *quadword* dalam mendeteksi plagirisme di antara dua dokumen teks. Pengujian ini bertujuan untuk mengetahui pendekatan token berbentuk manakah yang lebih baik dalam mendeteksi plagiarisme dokumen teks
3. Menguji apakah pendekatan token berbentuk *biword, triword* dan *quadword* telah memenuhi kebutuhan dasar algoritma pendeteksi plagiarisme dokumen teks seperti *whitespace insensitivity, noise surpression* dan *position independence.*
4. Menguji *threshold* kata yang dibentuk dalam mendeteksi plagiarisme di antara dua dokumen teks sehingga diperoleh suatu batas minimal banyak kata yang dapat dikatakan plagiarisme pada dokumen teks.

3.8. Kesimpulan dan Saran

Tahapan kesimpulan dan saran merupakan tahapan akhir dari penelitian ini. Tahapan ini berisi tentang kesimpulan dari hasil-hasil analisis aplikasi, perancangan aplikasi, implementasi aplikasi dan pengujian aplikasi yang telah dilakukan pada penelitian tugas akhir ini, sedangkan saran berisi masukan yang dapat dijadikan bahan penelitian ulang dalam meneliti dan membangun aplikasi pendeteksi plagiarisme dokumen teks yang lebih baik.

BAB IV

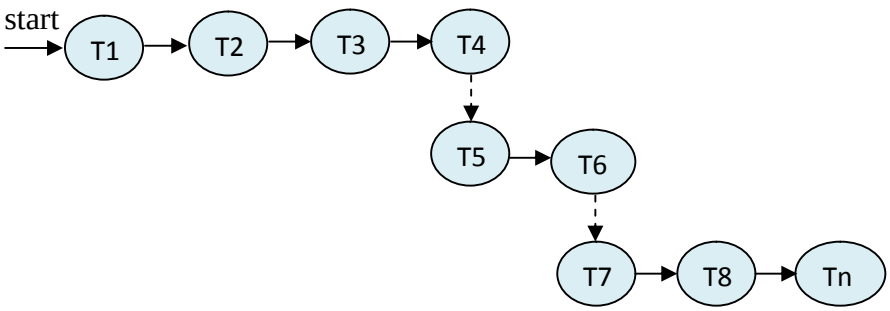
ANALISIS DAN PERANCANGAN

4.1. Analisis Pendekatan *K-gram*

Pendekatan *k-gram* merupakan metode yang digunakan untuk membentuk *substring* sepanjang *k* karakter atau kata dari sebuah *string*. Pendekatan ini, ditujukan untuk pembangkitan kata atau karakter dan digunakan untuk mengambil potongan-potongan karakter huruf atau kata dari sebuah teks secara kontinuitas yang dibaca dari awal dokumen teks hingga akhir dari dokumen teks. Pendekatan inilah yang digunakan untuk membentuk token kata menjadi *biword, triword* dan *quadword.*

Secara garis besar ada beberapa tahap dalam melakukan pendeteksian plagiarisme dokumen menggunakan pendekatan *k-gram*, diantaranya:

1. Melakukan *preprocessing* atau pembersihan teks.
2. Melakukan pemotongan teks menjadi *biword, triword* atau *quadword*.
3. *Intersect* potongan kata *biword, triword* atau *quadword* secara timbal-balik antara dokumen A ke B dan dokumen B ke dokumen A.
4. Temukan dan simpan pada daftar baru hasil irisan yang terurut berdasarkan indeks dari dokumen A dan dokumen B. Ada ilustrasi dalam memperoleh irisan yang terurut berdasarkan indeks dari dokumen A dan dokumen B dimana *T* = hasil irisan dari dokumen A dan dokumen B adalah sebagai berikut:



Gambar 4.1 Ilustrasi Menemukan Irisan terurut

5. Hitung kemiripan dokumen dari hasil irisan terurut dokumen A dan hasil irisan terurut dokumen B menggunakan persamaan 2.1.

Berikut contoh penerapan pendekatan *k-gram* berbentuk token *biword*, *triword* dan *quadword*:

```
“Jurusan Teknik Informatika Universitas Islam Negeri Sultan Syarif Kasim Riau telah berakreditasi B”
```

1. Lakukan pembersihan teks yaitu menghilangkan tanda baca dan karakter yang tidak relevan lainnya serta mengubah huruf kapital menjadi huruf kecil.
   Sehingga teks diatas menjadi sebagai berikut:
   ```
   jurusan teknik informatika universitas islam negeri sultan syarif kasim riau telah berakreditasi b
   ```

2. Lakukan pemotongan kata menjadi *biword*, *triword* atau *quadword*.
   Pemotongan kata berbentuk *biword*:
   ```
   jurusan teknik
   teknik informatika
   informatika universitas
   universitas islam
   islam negeri
   negeri suska
   suska riau
   riau telah
   telah berakreditasi
   berakreditasi b
   ```

   Pemotongan kata berbentuk *triword*:
   ```
   jurusan teknik informatika
   teknik informatika universitas
   informatika universitas islam
   universitas islam suska
   islam suska riau
   suska riau telah
   riau telah berakreditasi
   telah berakreditasi b
   ```

Pemotongan kata berbentuk *quadword*:

```
jurusan teknik informatika universitas
teknik informatika universitas islam
informatika universitas islam negeri
universitas islam negeri riau
islam negeri riau telah
negeri riau telah berakreditasi
riau telah berakreditasi b
```

3. Iriskan token yang telah terbentuk secara timbal-balik dari dokumen A ke B dan dokumen B ke A serta ambil hasil irisan yang terurut berdasarkan indeks. Ada langkah-langkah dalam mengiriskan dokumen secara timbal-balik dan mengambil hasil irisan terurut adalah sebagai berikut:

Misalkan hasil tokenisasi dokumen berbentuk *biword* seperti dibawah ini:

Tabel 4.1 Hasil Tokenisasi Dokumen

Dokumen A	Dokumen B
[0, jurusan teknik]	[0, peminat jurusan]
[1, teknik informatika]	[1, jurusan teknik]
[2, informatika universitas]	[2, teknik informatika]
[3, universitas islam]	[3, informatika universitas]
[4, islam negeri]	[4, universitas islam]
[5, negeri suska]	[5, islam negeri]
[6, suska riau]	[6, negeri suska]
[7, riau telah]	[7, suska riau]
[8, telah berakreditasi]	[8, riau meningkat]
[9, berakreditasi b]	[9, meningkat setiap]
	[10, setiap tahunnya]
	[11, tahunnya dikarenakan]
	[12, dikarenakan telah]
	[13, telah berakreditasi]
	[14, berakreditasi b]

Dari hasil tokenisasi dokumen diatas, kemudian iriskan dua dokumen tersebut secara timbal-balik antara dokumen A ke B dan dokumen B ke A sehingga diperoleh token *biword* yang sama dari dokumen A dan yang sama dari dokumen B. Untuk lebih jelasnya, berikut hasil irisan dari dokumen diatas dimana batas token terurut berdasarkan indeks yang diambil yaitu >=2:

Tabel 4.2. Hasil Irisan Terurut

Hasil irisan terurut dokumen A	Hasil irisan terurut dokumen B
Terurut 1: [0, jurusan teknik] [1, teknik informatika] [2, informatika universitas] [3, universitas islam] [4, islam negeri] [5, negeri suska] [6, suska riau] Terurut 2: [8, telah berakreditasi] [9, berakreditasi b]	Terurut 1: [1, jurusan teknik] [2, teknik informatika] [3, informatika universitas] [4, universitas islam] [5, islam negeri] [6, negeri suska] [7, suska riau] Terurut 2: [13, telah berakreditasi] [14, berakreditasi b]

4. Setelah melakukan *intersect* dan mengambil token terurut berdasarkan indeks di antara dua dokumen, langkah selanjutnya yaitu menghitung kemiripan dokumen berdasarkan *intersect* dari hasil terurut dan *union* dari token kata dokumen A dan B dengan menggunakan persamaan 2.1:

   Similaritas(d_i,d_j) = $\frac{|A(di) \cap B(dj)|}{|A(di) \cup B(dj)|} = \frac{9}{16} = 56\%$

4.2. Analisis Algoritma *Sieve of Erasthotenes*

Algoritma *sieve of erastothenes* merupakan sebuah algoritma yang biasanya digunakan untuk mempercepat pencarian bilangan prima. Konsep algoritma ini, yaitu menemukan bilangan prima dari sebuah daftar bilangan dan jika ketemu ambil bilangan tersebut, simpan pada daftar baru dan hapus dari daftar bilangan. Cara kerja algoritma inilah yang digunakan untuk menemukan dan

mempercepat pencarian pasangan hasil terurut di antara dokumen A dan dokumen B.

Ada langkah-langkah dalam penerapan algoritma ini sesuai Diagram alir 4.9 dalam menentukan pasangan hasil terurut dokumen A dan dokumen B yang terdapat pada Tabel 4.2 adalah sebagai berikut:

1. Misalkan hasil terurut 1 adalah 0 dan hasil terurut 2 adalah 1 berdasarkan indeks sehingga hasil terurut menjadi:
   Dokumen A = {0,1}
   Dokumen B = {0,1}
2. Langkah selanjutnya temukan hasil terurut yang sama di antara hasil terurut dokumen A dan hasil terurut dokumen B.
3. Setelah itu, cek indeks hasil terurut yang sama dari dokumen A dan dokumen B, apakah indeks hasil terurut antara dokumen A dokumen B saling berurutan serta memenuhi batas terurut yang telah ditentukan yaitu >= 2, jika memenuhi simpan pasangan hasil terurut yang sama tersebut pada *list* baru.
4. Hapus pasangan hasil terurut yang telah ditemukan dari daftar hasil terurut dokumen A dan hasil terurut dokumen B. Sebelum melakukan tahap ini, pastikan langkah nomor 2 dan langkah nomor 3 sudah dilakukan, sehingga daftar hasil terurut menjadi sebagai berikut:
   Dokumen A = {~~0~~,1}
   Dokumen B = {~~0~~,1}
   Sehingga daftar hasil terurut yang tersisa menjadi:
   Dokumen A = {1}
   Dokumen B = {1}
5. Ulangi langkah nomor 4 sampai semua pasangan hasil terurut ditemukan.
   Dokumen A = {~~1~~}
   Dokumen B = {~~1~~}
   Sehingga pasangan hasil irisan terurut dari dokumen A dan dokumen B adalah sebagai berikut:

Pasangan hasil irisan terurut pertama:

Dokumen A:

```
[0, jurusan teknik]
[1, teknik informatika]
[2, informatika universitas]
[3, universitas islam]
[4, islam negeri]
[5, negeri suska]
[6, suska riau]
```

Dokumen B:

```
[1, jurusan teknik]
[2, teknik informatika]
[3, informatika universitas]
[4, universitas islam]
[5, islam negeri]
[6, negeri suska]
[7, suska riau]
```

Pasangan hasil irisan terurut kedua:

Dokumen A:

```
[8, telah berakreditasi]
[9, berakreditasi b]
```

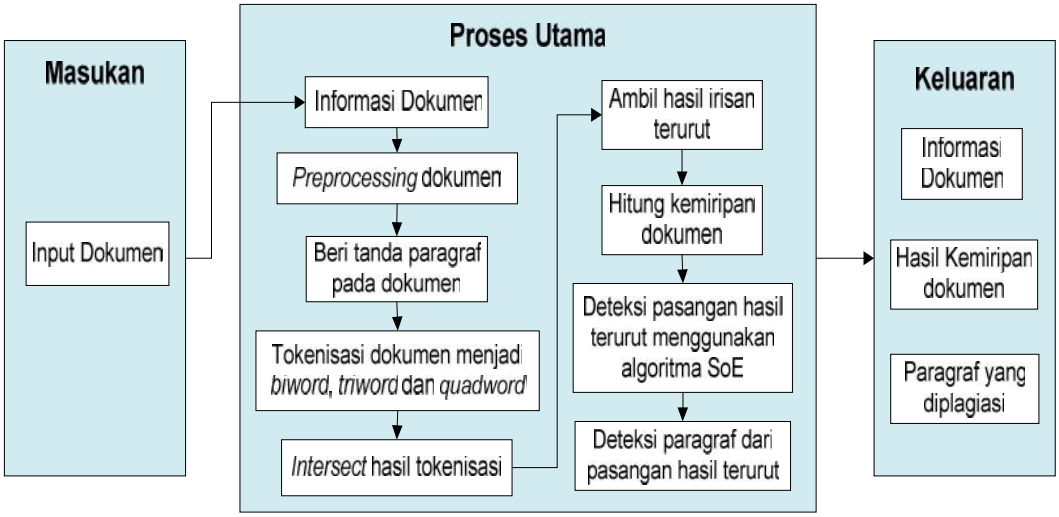
Dokumen B:

```
[13, telah berakreditasi]
[14, berakreditasi b]
```

4.3. Gambaran Umum Aplikasi Pendeteksi Plagiarisme

Gambaran umum aplikasi pendeteksi plagiarisme dokumen ditujukan untuk mengetahui tahapan demi tahapan proses jalanya aplikasi dalam melakukan pendeteksian plagiarisme dokumen. Tahapan tersebut, dimulai dari masukan, proses utama dan keluaran. Pada tahapan masukan, berisikan dokumen yang akan dideteksi tingkat kimiripannya. Kemudian, dokumen masukan tersebut diproses menggunakan pendekatan *k-gram* berbentuk token *biword, triword* dan *quadword* serta menggunakan algoritma *sieve of erasthotenes* untuk mempercepat pencarian paragraf yang diplagiasi. Setelah proses utama selesai dilakukan, aplikasi akan

memberikan keluaran berupa informasi dokumen, kemiripan dokumen serta paragraf yang diplagiasi. Untuk lebih jelasnya, berikut gambaran umum aplikasi pendeteksi plagiarisme.

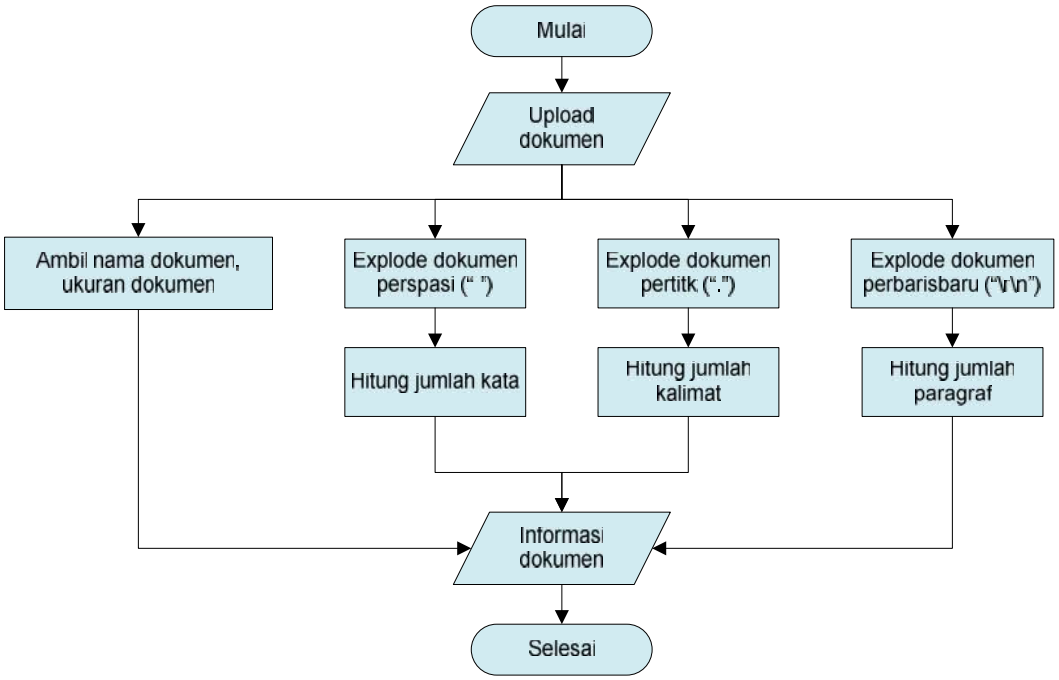


Gambar 4.2. Gambaran Umum Aplikasi Pendeteksi Plagiarisme

Berdasarkan Gambar 4.2 dapat diketahui bahwa rancang bangun aplikasi pendeteksian plagiarisme dokumen ini memiliki beberapa proses berupa masukan, proses utama dan keluaran yang dapat dijelaskan sebagai berikut:

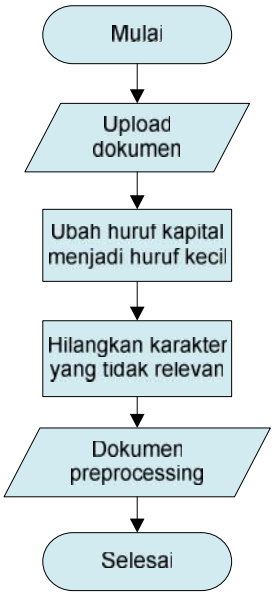
1. Masukan merupakan dokumen yang akan dideteksi tingkat kemiripannya serta menentukan kalimat yang sama pada dokumen tersebut.

2. Proses utama merupakan proses-proses utama yang terdapat pada rancang bangun aplikasi pendeteksi plagiarisme. Ada beberapa tahapannya adalah sebagai berikut:
   a. Proses utama dimulai dari baca dokumen *upload* yang akan dideteksi tingkat kemiripannya dan mengambil informasi dokumen berupa nama dokumen, ukuran dokumen, jumlah kata, jumlah kalimat dan jumlah paragraf pada dokumen. Untuk lebih jelasnya, ada diagram alir untuk memperoleh informasi dokumen adalah sebagai berikut:



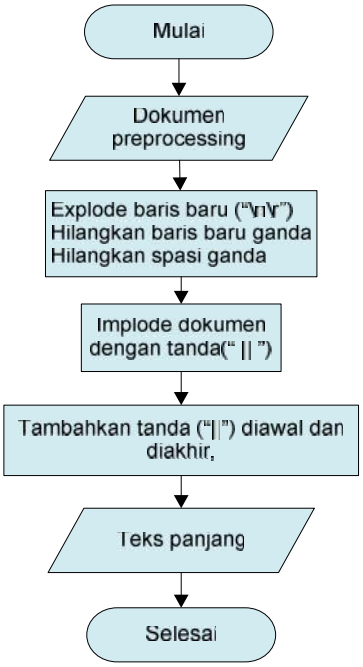
Gambar 4.3 Diagram Alir Informasi Dokumen

b. Melakukan *preprocessing* dokumen yaitu menghilangkan tanda baca, simbol dan karakter-karakter yang tidak relevan lainnya, misalnya: !,“”? dan lain-lain serta mengubah huruf kapital menjadi huruf kecil. Untuk lebih jelasnya, ada diagram alir dari tahapan pembersihan teks adalah sebagai berikut:



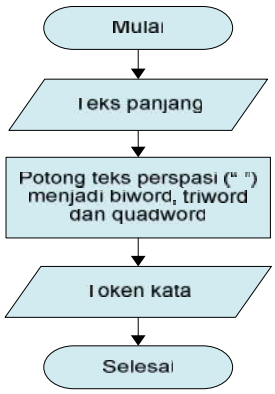
Gambar 4.4 Diagram Alir *Preprocessing* Dokumen

c. Mendeteksi paragraf dan menandainya, bertujuan untuk mendeteksi dan menampilkan paragraf yang diplagiasi. Pada tahap ini, juga dilakukan penghilangan *double newline* dan *double space*. Untuk lebih jelasnya, ada diagram alir untuk mendeteksi dan menandai paragraf adalah sebagai berikut:



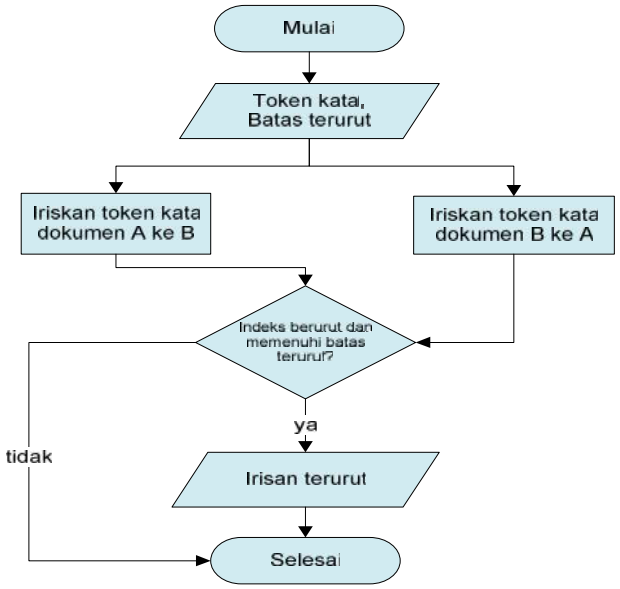
Gambar 4.5 Diagram Alir Penandaan Paragraf

d. Melakukan tokenisasi dokumen berbentuk token *biword*, *triword* dan *quadword*. Untuk lebih jelasnya, ada diagram alir untuk melakukan tokenisasi kata menjadi *biword*, *triword* dan *quadword* adalah sebagai berikut:



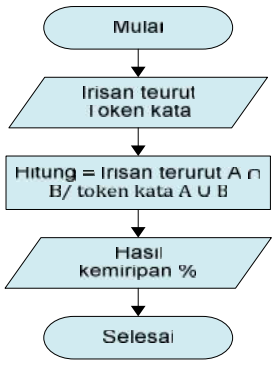
Gambar 4.6 Diagram Alir Tokenisasi Kata

e. Setelah melakukan token kata menjadi *biword, triword* atau *quadword*, kemudian token kata ini diiriskan timbal-balik antara dokumen A ke B dan dokumen B ke A dan mengambil irisan terurut sesuai batas terurut yang dimasukan. Untuk lebih jelasnya, ada diagram alir untuk melakukan *intersect* dokumen dan mengambil hasil irisan terurut adalah sebagai berikut:



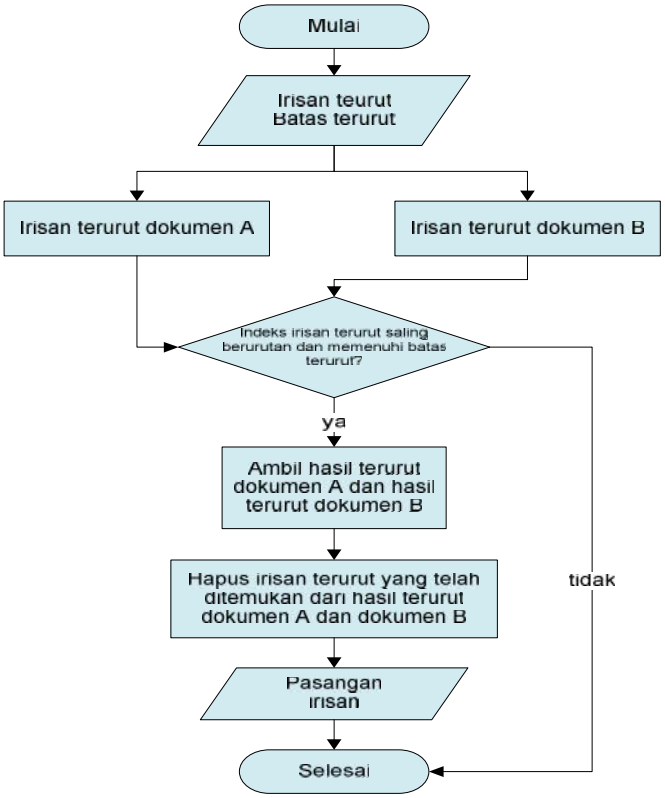
Gambar 4.7 Diagram Alir Irisan Terurut

f. Hasil irisan terurut yang telah ditemukan dihitung menggunakan *jaccard coefficient* untuk mengetahui tingkat kemiripan dokumen. Untuk lebih jelasnya, ada diagram alir untuk menghitung kemiripan dokumen adalah sebagai berikut:



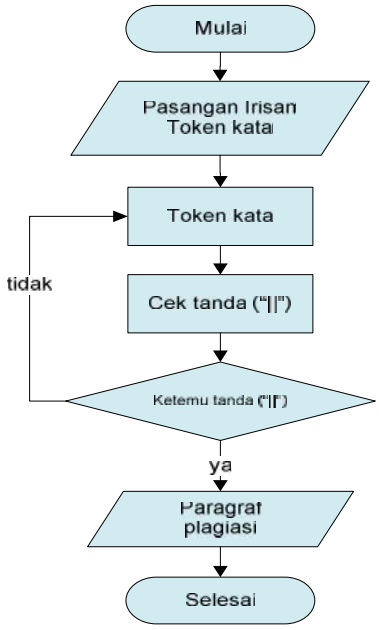
Gambar 4.8 Diagram Alir Mengukur Kemiripan Dokumen

g. Temukan pasangan hasil irisan terurut di antara dua dokumen teks. Untuk lebih jelasnya, ada diagram alir untuk menemukan pasangan irisan terurut di antara dua dokumen teks adalah sebagai berikut:



Gambar 4.9 Diagram Alir Pasangan Irisan Terurut

h. Terakhir, deteksi paragraf yang diplagiasi dari setiap pasangan hasil terurut berdasarkan paragraf yang telah ditandai sebelumnya. Untuk lebih jelasnya, ada diagram alir untuk menampilkan paragraf yang diplagiasi adalah sebagai berikut:



Gambar 4.10 Diagram Alir Paragraf yang Plagiasi

3. Keluaran merupakan hasil dari proses-proses utama yang terjadi pada aplikasi. Keluaran pada aplikasi ini dapat berupa informasi dokumen yaitu nama dokumen, ukuran dokumen, jumlah kata, jumlah kalimat, jumlah paragraf serta waktu proses, hasil kemiripan dan paragraf yang diplagiasi.

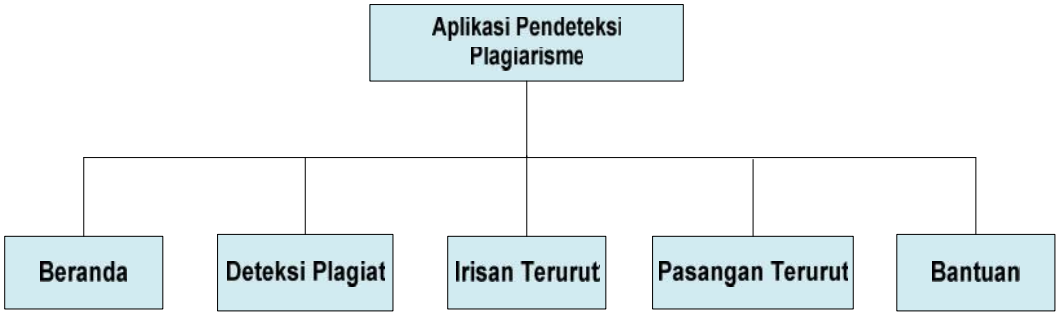
4.4. Perancangan Aplikasi

Setelah dilakukan beberapa tahapan dalam analisis aplikasi, maka dapat dilakukan beberapa perancangan untuk aplikasi pendeteksi palgiarisme ini. Perancangan-perancangan yang akan dijelaskan dalam laporan ini meliputi perancangan antarmuka aplikasi serta perancangan struktur menu aplikasi.

4.4.1. Perancangan Struktur Menu

Dalam membangun aplikasi pendeteksi plagiarisme dokumen diperlukan susunan daftar pilihan atau menu yang mudah untuk dimegerti sehingga pengguna yang belum terbiasa dapat dengan cepat beradaptasi dalam menggunakan aplikasi ini. Selain itu, pengguna juga akan dihadapkan pada berbagai alternatif menu yang ada. Dalam menentukan pilihannya, pengguna dapat menggunakan tombol yang

tersedia dan setiap pilihan akan menghasilkan tanggapan atau jawaban tertentu. Aplikasi yang akan dibangun ini, memiliki menu-menu yang digambarkan pada bagan di bawah ini:



Gambar 4.11 Struktur Menu

Menu beranda pada aplikasi pendeteksi plagiarisme ini akan menampilkan halaman utama dari aplikasi yang berisi sedikit penjelasan tentang plagiarisme dan algoritma yang digunakan pada aplikasi ini. Selanjutnya, menu deteksi plagiat merupakan menu yang digunakan untuk mengekspor dokumen ke format *plain text*, menguji dua dokumen teks, menampilkan hasil kemiripan dokumen serta paragraf yang diplagiasi. Kemudian, menu irisan terurut merupakan menu yang berisikan hasil terurut dari irisan dua dokumen teks. Setelah itu, menu pasangan terurut merupakan menu yang berisikan pasangan irisan terurut yang telah terbentuk dari dua dokumen teks. Terakhir, menu bantuan merupakan menu yang berisikan penjelasan tentang langkah-langkah menggunakan aplikasi pendeteksi plagiarisme dokumen ini.

4.4.2. Perancangan Antarmuka

Antarmuka aplikasi merupakan suatu sarana pengembangan aplikasi yang ditujukan untuk mempermudah pengguna berkomunikasi dengan aplikasi yang dibangun. Hal utama yang harus diperhatikan pada antarmuka meliputi tampilan yang baik, mudah dipahami dan tombol-tombol yang familiar. Rancang bangun dari aplikasi pendeteksi plagiarisme dokumen ini memiliki lima menu, yaitu menu beranda, deteksi plagiat, Irisan terurut, pasangan terurut dan menu bantuan. Menu beranda merupakan menu yang akan menampilkan halaman utama aplikasi. Menu

deteksi plagiat merupakan menu yang digunakan untuk mengekspor dokumen ke format *plain text*, menguji dua dokumen teks, menampilkan hasil kemiripan dokumen serta paragraf yang diplagiasi.

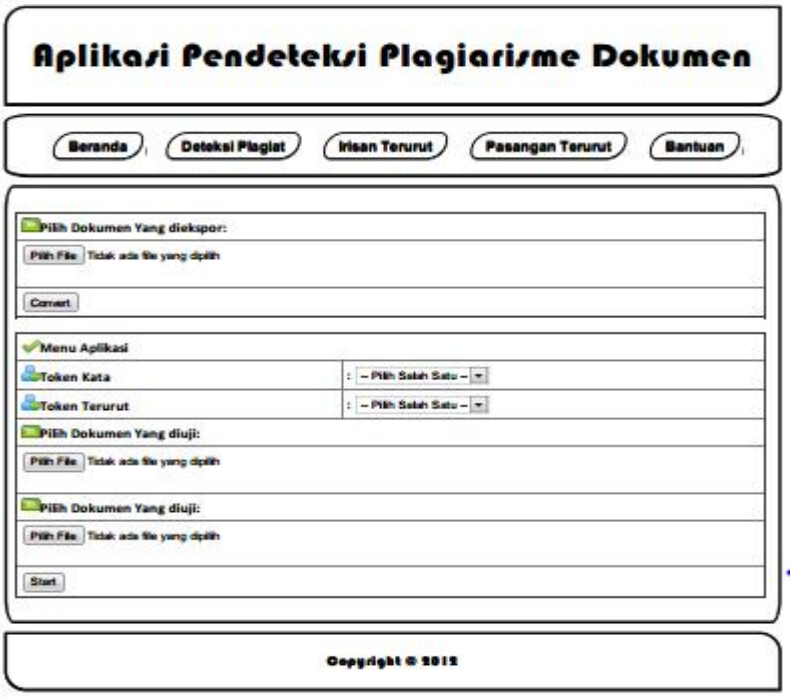
Pada aplikasi pendeteksi plagiarisme dokumen teks ini, halaman utama berisi sedikit tentang defenisi dari plagiarisme dokumen dan algoritma yang digunakan, yang lebih jelasnya dapat dilihat pada Gambar 4.12 berikut:

Gambar 4.12 Antarmuka Halaman Utama

Tabel 4.4 Spesifikasi *Function Key* atau Objek Tampilan Menu Utama

Nama Objek	Jenis	Keterangan
Beranda	MenuBar	*Form* untuk halaman utama aplikasi
Deteksi Plagiat	MenuBar	Berisikan *form* untuk mendeteksi plagiarisme dokumen
Token terurut	MenuBar	*Form* untuk menampilkan array terurut dari hasil *intersect*
Hasil Pengujian	MenuBar	*Form* untuk menampilkan pengujian-pengujian yang telah dilakukan
Tentang	MenuBar	Halaman yang berisikan tentang algoritma yang digunakan dalam membangun aplikasi

Berdasarkan Gambar 4.12 dapat dijelaskan bahwa untuk memulai penggunaan aplikasi ini seorang pengguna akan dihadapkan pada halaman utama dan untuk melakukan pendeteksian kemiriapan dua dokumen teks seorang pengguna dapat memilih menu deteksi plagiat. Setelah memilih menu ini, aplikasi akan menampilkan *form* untuk mengekspor dokumen ke format *plain text* dan *form* untuk mendeteksi kemiripan dua dokumen teks, dimana tampilannya dapat dilihat pada Gambar 4.13.

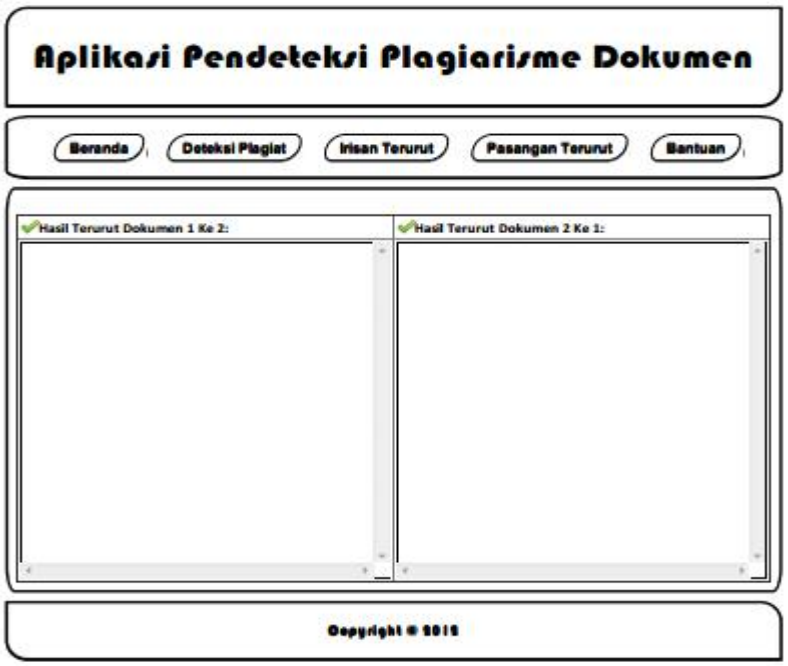


Gambar 4.13 Antarmuka Halaman Deteksi Plagiat

Tabel 4.5 Spesifikasi *Function Key* atau Objek Tampilan Deteksi Plagiat

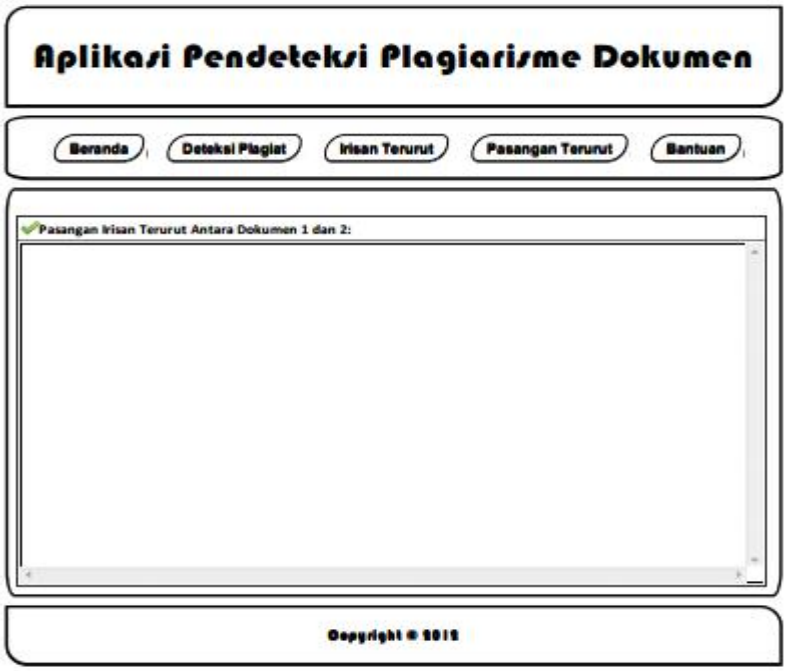
Nama Objek	Jenis	Keterangan
Pilih file	Button	*Browse* dokumen yang akan diekspor ke *plain text*
Convert	Button	Melakukan eksekusi ekspor dokumen ke *plain text*
Token	Select	Berisikan jumlah token yang akan digunakan
Token terurut	Select	Berisikan batas token terurut yang akan digunakan
Pilih file	Button	*Browse* dokumen yang akan diuji
Pilih file	Button	*Browse* dokumen yang akan diuji
Start	Button	Melakukan pendeteksian plagiarisme dokumen

Selanjutnya, pengguna juga dapat melihat irisan terurut yang dihasilkan dari *intersect* dua dokumen teks, tampilannya dapat dilihat pada Gambar 4.14.



Gambar 4.14 Antarmuka Irisan Terurut

Selain itu, pengguna juga dapat melihat pasangan irisan terurut dari dua dokumen teks, tampilannya dapat dilihat pada Gambar 4.15.



Gambar 4.15 Antarmuka Pasangan Terurut

BAB V

IMPLEMENTASI DAN PENGUJIAN

5.1. Tahapan Implementasi

Tahapan implementasi merupakan tahapan dimana suatu aplikasi atau perangkat lunak yang telah dianalisis, dirancang dan selanjutnya akan direalisasikan sebagai serangkaian program serta diuji kelayakannya. Sehingga, akan diketahui bahwa aplikasi yang dibuat telah sesuai dengan tujuan yang diinginkan dan dapat dioperasikan sebagaimana mestinya. Berikut ini penjelasan tentang pengimplementasian aplikasi pendeteksi plagiarisme dokumen berdasarkan analisis dan perancangan yang telah dilakukan sebelumnya.

5.1.1. Batasan Implementasi

Aplikasi pendeteksi plagiarisme dokumen yang dibangun pada tugas akhir ini memiliki batasan implementasi sebagai berikut:

1. Bahasa pemrograman yang digunakan dalam pengimplementasian aplikasi ini yaitu Php pada sistem operasi *Microsoft Windows 7 Ultimate*.
2. Dokumen masukan yang dapat dideteksi yaitu dokumen yang berformat *plain text (txt* dan *dtxt)*.

5.1.2. Lingkungan Operasional

Pengimplementasian aplikasi pendeteksi plagiarisme dokumen ini dibagi kedalam dua komponen yaitu perangkat keras dan perangkat lunak, berikut ini adalah lingkungan operasional yang digunakan dalam pengimplementasian aplikasi:

1. Perangkat keras

   Processor : *Intel(R) Core(TM) i3 CPU M330 @ 2.13GHz*

   Memori (RAM) : 2,00 GB

Harddisk : 320 GB

2. Perangkat Lunak

   Sistem Operasi : *Windows 7 Ultimate*

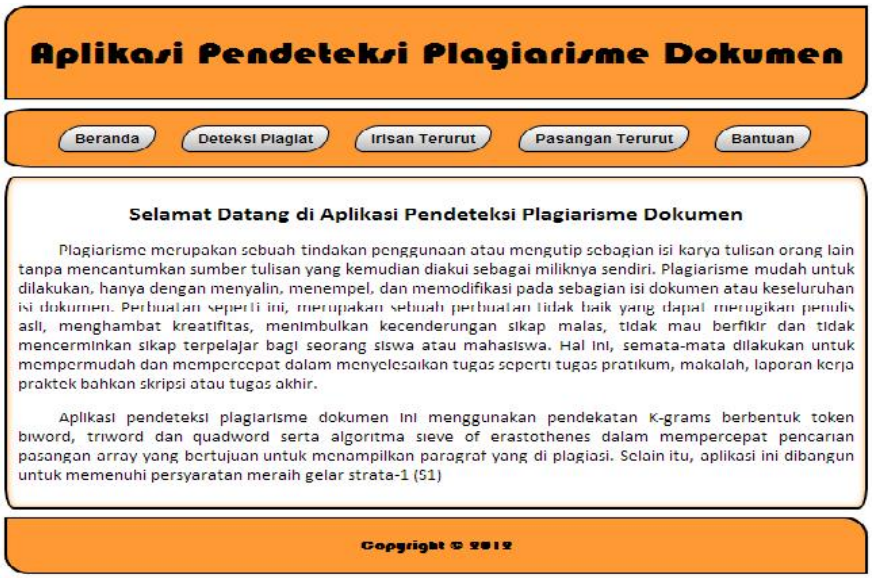
   Bahasa Pemrograman : PHP versi 5.2.6

   Tools Perancangan : Notepad++

5.1.3. Implementasi Antarmuka Aplikasi

Setelah tahap analisis dan perancangan selesai dilakukan, maka dilanjutkan dengan tahap implementasi aplikasi dari hasil analisis yang telah diperoleh dan mengimplementasikan hasil perancangan antarmuka yang telah dibuat. Berikut ini akan dijelaskan mengenai hasil implementasi dari rancang bangun aplikasi pendeteksi plagiarisme dokumen ini, dimana pada aplikasi pendeteksi plagiarisme dokumen ini memiliki lima menu, yaitu menu beranda, menu deteksi plagiat, menu irisan terurut, menu pasangan terurut dan menu bantuan. Menu beranda merupakan menu yang akan menampilkan halaman utama aplikasi. Menu deteksi plagiat merupakan menu yang digunakan untuk mengekspor dokumen ke format *plain text*, mendeteksi kemiripan dua dokumen teks, menampilkan hasil kemiripan dokumen serta paragraf yang diplagiasi. Berikut implementasi aplikasi pendeteksi plagiarisme dokumen ini sesuai dengan menu yang ada pada aplikasi:

1. Implementasi Antarmuka Menu Beranda



Gambar 5.1 Hasil Implementasi Antarmuka Menu Beranda

Antarmuka pada Gambar 5.1 merupakan tampilan yang akan muncul pertama sekali ketika pengguna menjalankan aplikasi ini.

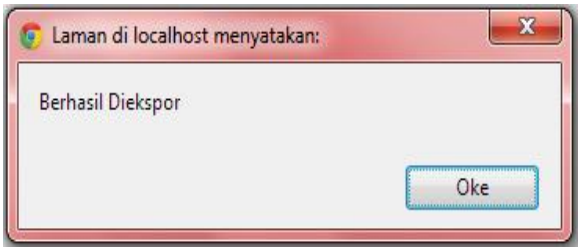
2. Implementasi Antarmuka Menu Deteksi Plagiat

Gambar 5.2 Hasil Implementasi Antarmuka Menu Deteksi Plagiat

Pada Gambar 5.2 merupakan tampilan menu deteksi plagiat dimana pada menu ini terdapat *form* yang digunakan untuk mendeteksi kemiripan dua dokumen teks. Selain itu, pengguna juga dapat mengekspor dokumen berformat *doc* dan *docx* ke *plain text* agar dapat dieksekusi oleh aplikasi ini.

3. Implementasi *Alert* pada proses ekspor dokumen

Gambar 5.3 adalah *alert* ketika mengekspor dokumen *doc* dan *docx* menjadi *plain text*, *alert* ini akan memberitahu bahwa dokumen berhasil diekspor menjadi *plain text*.



Gambar 5.3 *Alert* Ekspor Dokumen

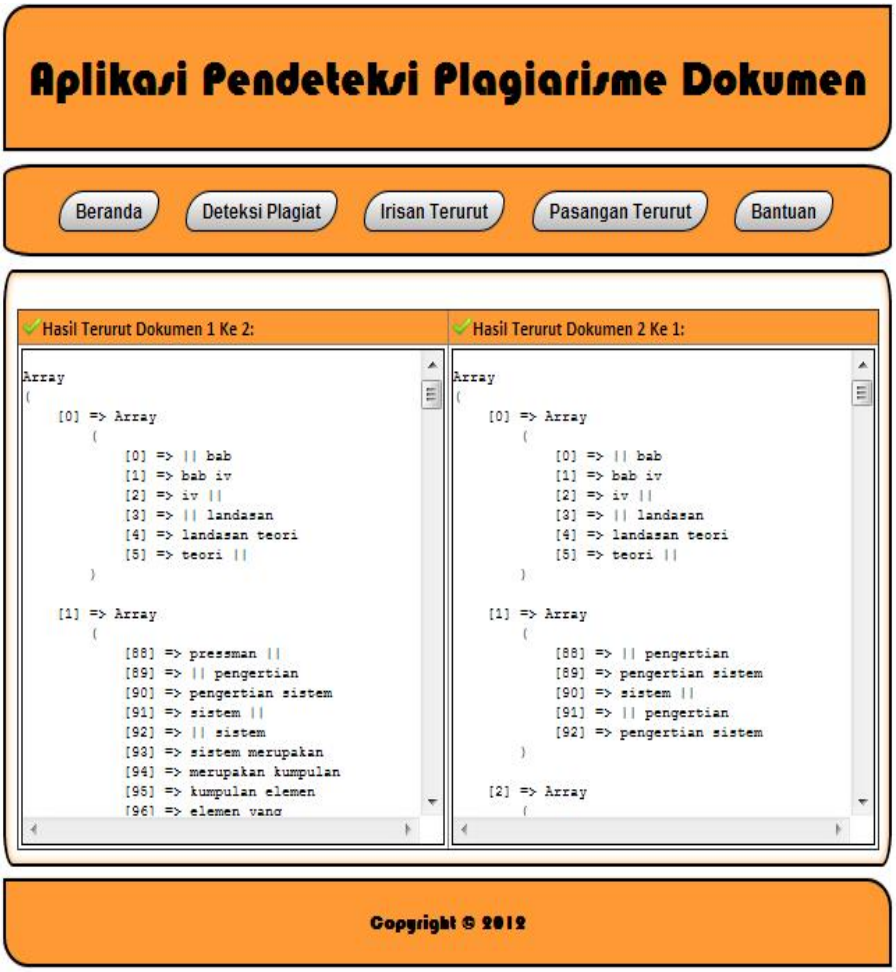
4. Implementasi tampilan hasil deteksi kemiripan dokumen

Gambar 5.4 merupakan tampilan antarmuka yang dihasilkan aplikasi ketika pengguna telah melakukan pendeteksian kemiripan diantara dua dokumen teks dengan memasukan parameter-parameter yang terdapat pada Gambar 5.2. Hasil yang ditampilkan oleh aplikasi ini yaitu informasi dokumen teks, kemiripan dokumen teks dan paragraf yang diplagiasi.

Gambar 5.4 Hasil Implementasi Antarmuka Hasil Deteksi Dokumen

5. Implementasi Antarmuka Menu Irisan Terurut

Antarmuka menu irisan terurut merupakan antarmuka aplikasi yang digunakan untuk melihat hasil irisan terurut di antara dua dokumen teks. Untuk lebih jelasnya berikut tampilan hasil implementasi antarmuka menu irisan terurut:



Gambar 5.5 Hasil Implementasi Antarmuka Menu Irisan Terurut

6. Implementasi Antarmuka Menu Pasangan Terurut

Antarmuka menu pasangan terurut merupakan antarmuka aplikasi yang digunakan untuk melihat pasangan irisan terurut dari dua dokumen teks. Untuk lebih jelasnya berikut tampilan hasil implementasi antarmuka menu pasangan terurut:

Gambar 5.7 Hasil Implementasi Antarmuka Menu Pasangan Terurut

7. Implementasi Antarmuka Menu Bantuan

Gambar 5.8 merupakan hasil implementasi antarmuka menu bantuan yang memuat tentang langkah-langkah menggunakan aplikasi ini. Untuk lebih jelasnya berikut tampilan hasil implementasi antarmuka menu pasangan terurut:

Gambar 5.8 Hasil Implementasi Antarmuka Menu Bantuan

5.2. Pengujian aplikasi

Setelah tahap implementasi selesai, maka dilanjutkan dengan tahap pengujian dari implementasi yang telah dibuat. Pengujian aplikasi dilakukan bertujuan untuk menjamin aplikasi yang dibangun sesuai dengan hasil analisis dan perancangan sehingga dapat dibuat satu kesimpulan akhir.

5.2.1. Rencana Pengujian

Rencana pengujian yang akan dilakukan adalah sebagai berikut:

1. Jumlah dokumen yang diuji sebanyak 3 dokumen dengan kategori tingkat kemiripan rendah, sedang dan tinggi.
2. Menguji aplikasi dengan *white box* bertujuan untuk mengetahui apakah algoritma yang telah diterapkan telah berfungsi sesuai dengan yang diharapkan.
3. Menguji hipotesa token berbentuk *biword, triword* dan *quadword* dalam mendeteksi plagirisme di antara dua dokumen teks. Pengujian ini bertujuan untuk mengetahui pendekatan token berbentuk manakah yang lebih baik dalam mendeteksi plagiarisme dokumen teks
4. Menguji apakah pendekatan token berbentuk *biword, triword* dan *quadword* telah memenuhi kebutuhan dasar algoritma pendeteksi plagiarisme dokumen teks seperti *whitespace insensitivity, noise surpression* dan *position independence.*
5. Menguji *threshold* kata yang dibentuk dalam mendeteksi plagiarisme di antara dua dokumen teks sehingga diperoleh suatu batas minimal banyak kata yang dapat dikatakan plagirisme pada dokumen teks.

5.2.1.1.Pengujian Aplikasi dengan *Whitebox*

Pengujian aplikasi dilakukan untuk memeriksa kinerja proses-proses utama pada aplikasi yang telah diimplementasikan. Tujuan utama dari pengujian aplikasi ini yaitu memastikan bahwa proses-proses utama yang telah diimplementasikan telah berfungsi sesuai dengan yang diharapkan. Salah satu

metode pengujian jenis ini dikenal dengan pengujian *whitebox.* Ada hasil dari pengujian yang telah dilakukan dengan menggunakan metode ini adalah sebagai berikut:

1. Proses Informasi dokumen

Pengujian proses informasi dokumen ini bertujuan untuk memastikan bahwa proses informasi dokumen yang telah diimplementasikan telah berfungsi dan sesuai dengan yang diharapkan yaitu menghitung jumlah kata, jumlah kalimat dan jumlah paragraf yang terdapat pada dokumen teks. Ada tahapan proses untuk memperoleh informasi dokumen teks ini dapat dilihat pada Diagram Alir 4.3.

Tabel 5.1. Hasil Pengujian Proses Informasi Dokumen

No	Objek Pengujian	Hasil yang diharapkan	Hasil
1	Proses menghitung jumlah kata pada dokumen teks	Dokumen teks yang telah dimasukan dibaca dan dihitung jumlah kata yang terdapat pada dokumen tersebut	Benar
2	Proses menghitung jumlah kalimat pada dokumen teks	Dokumen teks yang telah dimasukan dibaca dan dihitung jumlah kalimat yang terdapat pada dokumen tersebut	Benar
3	Proses menghitung jumlah paragraf pada dokumen teks	Dokumen teks yang telah dimasukan dibaca dan dihitung jumlah paragraf yang terdapat pada dokumen tersebut	Benar

Ada hasil pengujian informasi dokumen yang ditampilkan oleh aplikasi pendeteksi plagiarisme ini dapat dilihat pada Gambar 5.9.

Gambar 5.9 Hasil Pengujian Informasi Dokumen

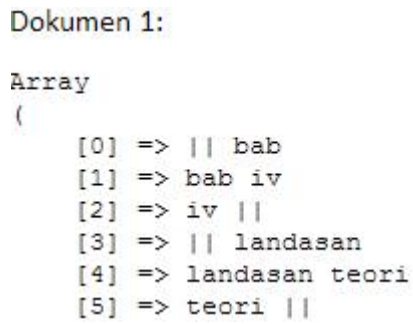
2. Proses Tokenisasi Dokumen

Pengujian proses tokenisasi kata ini bertujan untuk memastikan bahwa proses ini telah berfungsi dan sesuai dengan yang dinginkan yaitu memotong dokumen teks yang telah dimasukan menjadi token berbentuk *biword*, *triword* atau *quadword*. Untuk tahapan proses dalam melakukan tokenisasi kata ini dapat dilihat pada Diagram Alir 4.6.

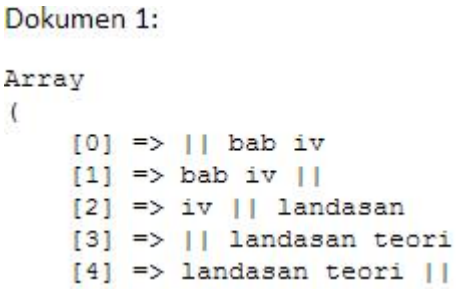
Tabel 5.2. Hasil Pengujian Proses Tokenisasi Dokumen

No	Objek Pengujian	Hasil yang diharapkan	Hasil
1	Proses tokenisasi dokumen teks menjadi *biword*	Dokumen teks yang telah dimasukan dipotong menjadi token berbentuk *biword*	Benar
2	Proses tokenisasi dokumen teks menjadi *triword*	Dokumen teks yang telah dimasukan dipotong menjadi token berbentuk *triword*	Benar
3	Proses tokenisasi dokumen teks menjadi *quadword*	Dokumen teks yang telah dimasukan dipotong menjadi token berbentuk *quadword*	Benar

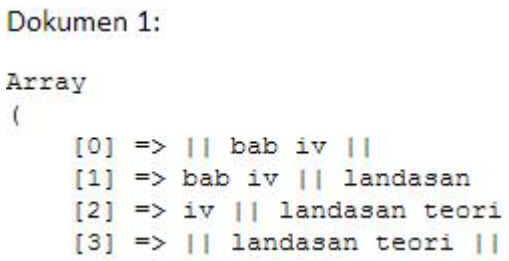
Ada hasil tokenisasi dokumen teks berbentuk *biword*, *triword* dan *quadword* yang ditampilkan oleh aplikasi adalah sebagai berikut:



Gambar 5.10 Tokenisasi Berbentuk *Biword*



Gambar 5.11 Tokenisasi Berbentuk *Triword*



Gambar 5.11 Tokenisasi Berbentuk *Quadword*

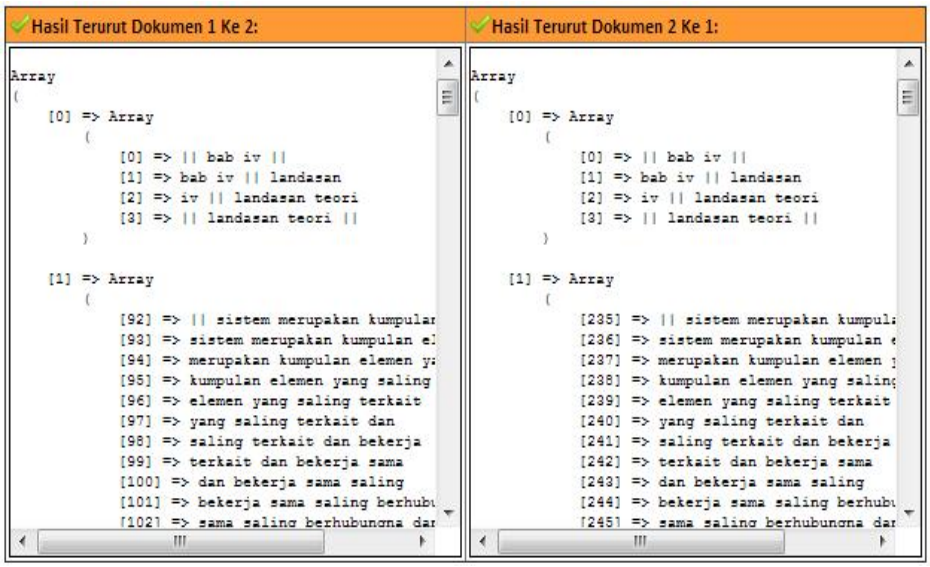
3. Proses Menemukan Irisan Terurut

Pengujian proses menemukan irisan terurut ini bertujan untuk memastikan bahwa proses ini telah berfungsi dan sesuai dengan yang dinginkan yaitu menemukan hasil irisan terurut berdasarkan indeks di antara dua dokumen teks yang telah dimasukan. Ada tahapan proses menemukan irisan terurut di antara dua dokumen teks dapat dilihat pada Diagram Alir 4.7.

Tabel 5.3. Hasil Pengujian Proses Menemukan Irisan Terurut

No	Objek Pengujian	Hasil yang diharapkan	Hasil
1	Proses proses menemukan irisan terurut sesuai batas minimal terurut yang dimasukan	Menampilkan hasil irisan terurut di antara dua dokumen teks sesuai batas minimal terurut yang dimasukan	Benar

Ada hasil irisan terurut di antara dua dokumen teks yang ditampilkan oleh aplikasi, dimana token yang digunakan *quadword* dan batas terurut yang dimasukan >=2 dapat dilihat pada Gambar 5.12.



Gambar 5.12 Hasil Pengujian Irisan Terurut

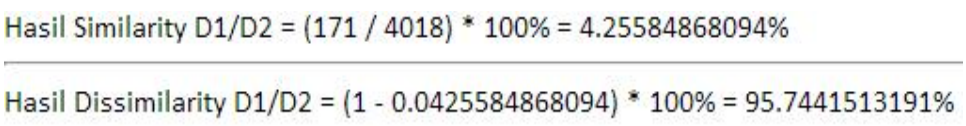
4. Proses Mengukur Kemiripan dokumen

Pengujian proses mengukur kemiripan dokumen ini bertujan untuk memastikan bahwa proses ini telah berfungsi dan sesuai dengan yang dinginkan yaitu aplikasi dapat mengukur kemiripan dokumen menggunakan persamaan 2.1. Ada tahapan proses dalam mengukur kemiripan dokumen teks dapat dilihat pada Diagram Alir 4.8.

Tabel 5.4. Hasil Pengujian Proses Mengukur Kemiripan Dokumen

No	Objek Pengujian	Hasil yang diharapkan	Hasil
1	Proses mengukur kemiripan dokumen	Menampilkan kemiripan dokumen teks yang diukur dari jumlah irisan terurut dan jumlah gabungan token di antara dua dokumen teks	Benar

Gambar 5.12. merupakan hasil perhitungan kemiripan dokumen teks yang ditampilkan aplikasi pendeteksi plagiarisme dokumen ini.



Gambar 5.12 Hasil Pengujian Proses Mengukur Kemiripan Dokumen

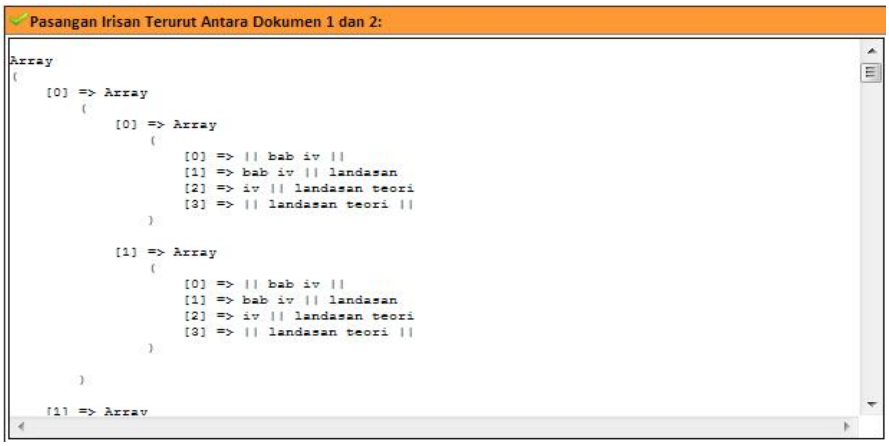
5. Proses Menemukan Pasangan Irisan Terurut

Pengujian proses menemukan pasangan irisan terurut ini bertujan untuk memastikan bahwa proses ini telah berfungsi dan sesuai dengan yang dinginkan yaitu menemukan pasangan irisan terurut dari hasil irisan terurut yang sama di antara dua dokumen teks. Ada tahapan proses menemukan pasangan irisan terurut di antara dua dokumen teks dapat dilihat pada Diagram Alir 4.9.

Tabel 5.5. Hasil Pengujian Proses Menemukan Pasangan Irisan Terurut

No	Objek Pengujian	Hasil yang diharapkan	Hasil
1	Proses menemukan pasangan irisan terurut	Menampilkan pasangan irisan terurut di antara dua dokumen teks sesuai batas minimal terurut yang dimasukan	Benar

Ada hasil pengujian proses menemukan pasangan irisan terurut di antara dua dokumen teks yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.13.



Gambar 5.13. Hasil Pengujian Pasangan Irisan Terurut

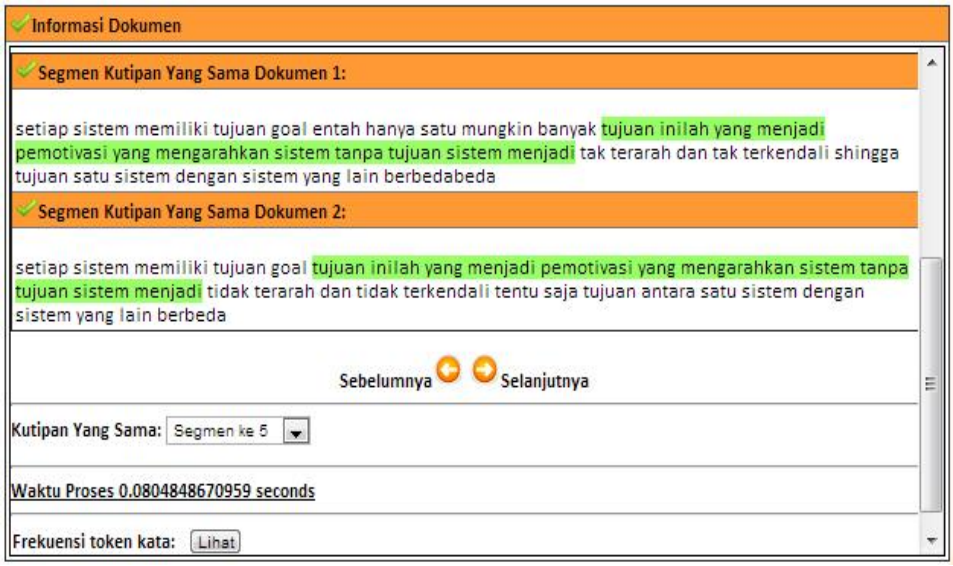
6. Proses Menemukan Paragraf yang Diplagiasi

Pengujian proses menemukan paragraf yang diplagiasi ini bertujan untuk memastikan bahwa proses ini telah berfungsi dan sesuai dengan yang dinginkan yaitu menemukan paragraf yang diplagiasi berdasarkan pasangan irisan terurut di antara dua dokumen teks. Ada tahapan proses menemukan paragraf yang diplagiasi di antara dua dokumen teks dapat dilihat pada Diagram Alir 4.10.

Tabel 5.5. Hasil Pengujian Proses Menemukan Paragraf yang Diplagiasi

No	Objek Pengujian	Hasil yang diharapkan	Hasil
1	Proses menemukan irisan terurut sesuai batas minimal terurut yang dimasukan	Dokumen teks yang telah dimasukan diiriskan dan menampilkan hasil irisan terurut di antara dua dokumen teks sesuai batas minimal terurut yang dimasukan	Benar

Gambar 5.14. merupakan tampilan paragraf yang diplagiasi di antara dua dokumen teks yang ditampilkan aplikasi pendeteksi plagiarisme dokumen ini.



Gambar 5.14 Hasil Pengujian Menemukan Paragraf yang Diplagiasi

5.2.1.2.Pengujian Hipotesa *Biword, Triword* dan *Quadword*

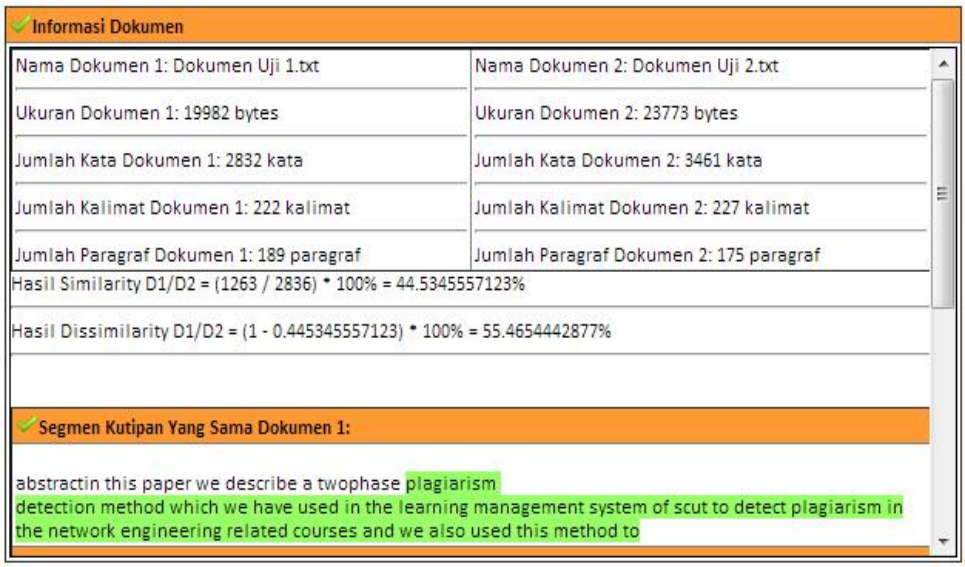
Pengujian hipotesa token berbentuk *biword, triword* dan *quadword* merupakan pengujian yang dilakukan untuk mendapatkan konfigurasi yang paling baik dalam mendeteksi kemiripan di antara dua dokumen teks. Pengujian ini dilakukan terhadap sebuah *paper* dari seorang penulis Y dengan konfigurasi awal panjang kata yang dianggap sama yaitu lima kata tiap-tiap token. Ada beberapa pengujian yang dilakukan dengan berbagai konfigurasi yang ada pada aplikasi pendeteksi palagiarisme ini adalah sebagai berikut:

1. Pengujian I

Pada pengujian ini digunakan konfigurasi aplikasi pendeteksi plagirisme dokumen teks ini adalah sebagai berikut:

a. Menggunakan token *biword*
b. Menggunakan batas *(threshold)* token terurut >= 4

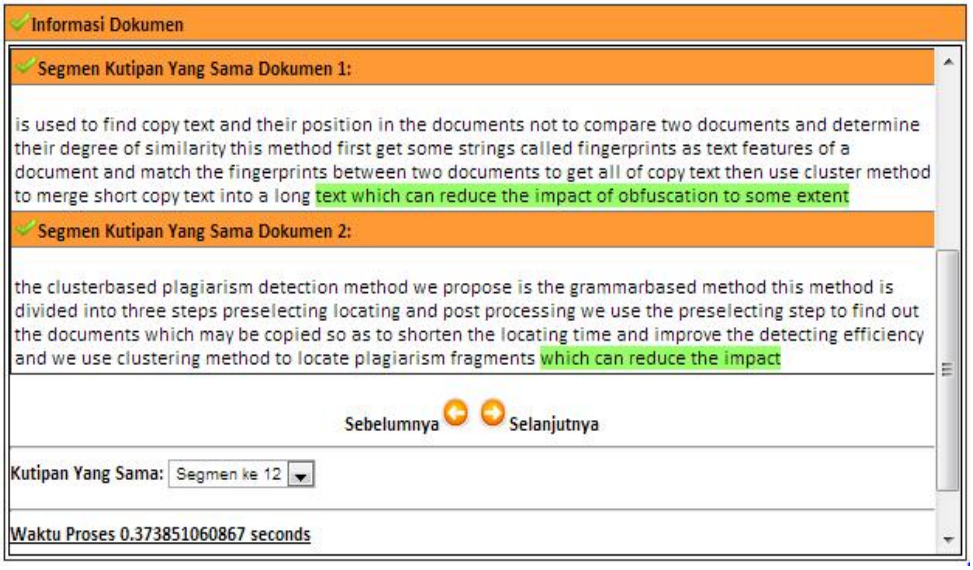
Ada hasil pengujian dari konfigurasi diatas yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.15.



Gambar 5.15 Hasil Kuantitatif Pengujian I

Dari Gambar 5.15 dapat dilihat bahwa dengan menggunakan token berbentuk *biword* dan batas token terurut >= 4 menghasilkan tingkat kemiripan yang cukup tinggi yaitu 44,53% dengan waktu proses 0.375 detik.

Dilihat dari segi kutipan sama yang ditampilkan oleh aplikasi terdapat cukup banyak kecocokan token *biword* yang bersifat kebetulan *(coincidental).* Hal ini terjadi karena pada saat pencocokan sebuah token *biword* di antara dua dokumen teks token tersebut sama akan tetapi, dalam pengambilan irisan terurut berdasarkan indeks pada salah satu dokumen hasil irisan tersebut tidak terurut berdasarkan indeks. Untuk lebih jelasnya, ada kutipan sama yang ditampilkan aplikasi berdasarkan kombinasi pada pengujian ini dapat dilihat pada Gambar 5.16.



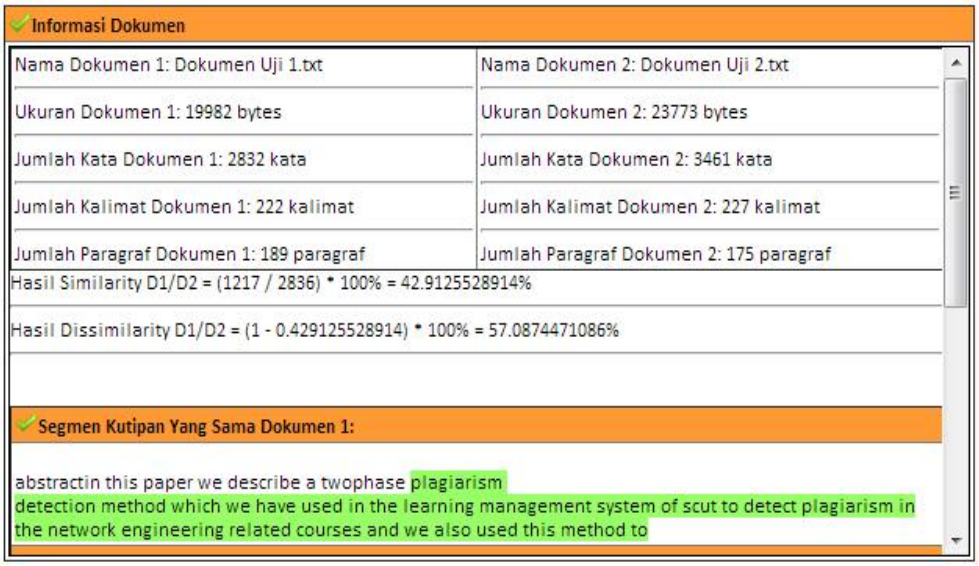
Gambar 5.16 Hasil Kualitatif Pengujian I

2. Pengujian II

Pada pengujian ini digunakan konfigurasi aplikasi pendeteksi plagirisme dokumen teks ini adalah sebagai berikut:

a. Menggunakan token *biword*
b. Menggunakan batas *(threshold)* token terurut >= 5

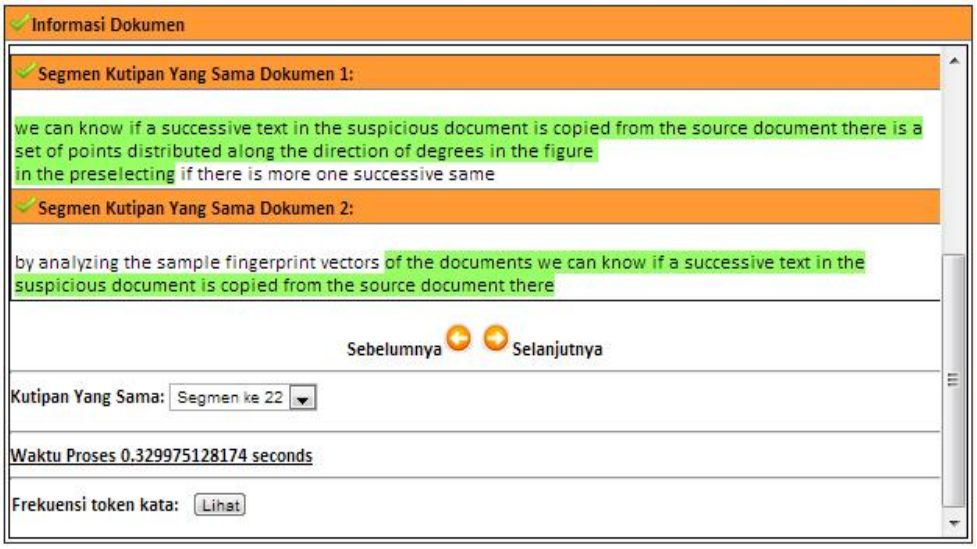
Ada hasil pengujian dari konfigurasi diatas yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.17.



Gambar 5.17 Hasil Kuantitatif Pengujian II

Dari Gambar 5.17 dapat dilihat bahwa dengan menggunakan token berbentuk *biword* dan batas token terurut >= 5 menghasilkan tingkat kemiripan yaitu 42,91% dengan waktu proses 0.327 detik. Berkurangnya tingkat kemiripan dokumen teks dari konfigurasi sebelumnya karena batas minimal kata yang dianggap sama ditingkatkan menjadi enam kata sehingga irisan terurut yang terbentuk lebih sedikit dari sebelumnya.

Dilihat dari segi kutipan sama yang ditampilkan oleh aplikasi, pada konfigurasi ini menemukan masih adanya kecocokan token *biword* yang bersifat kebetulan *(coincidental)* walaupun telah meningkatkan batas minimal kata yang dianggap sama. Ada kutipan sama yang ditampilkan aplikasi ini berdasarkan konfigurasi pada pengujian ini dapat dilihat pada Gambar 5.18.



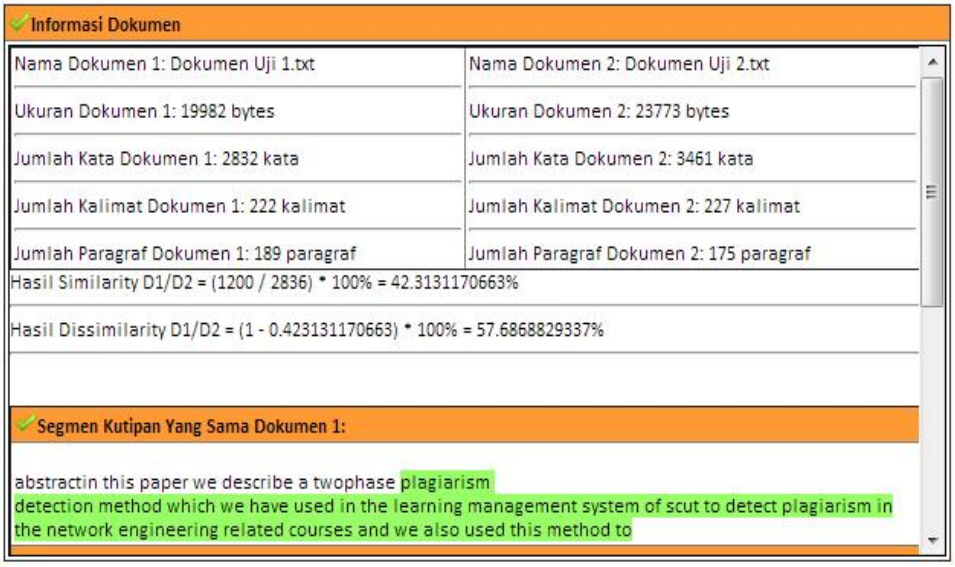
Gambar 5.18 Hasil Kualitatif Pengujian II

3. Pengujian III

Pada pengujian ini digunakan konfigurasi aplikasi pendeteksi plagirisme dokumen teks ini adalah sebagai berikut:

a. Menggunakan token *biword*
b. Menggunakan batas *(threshold)* token terurut >= 6

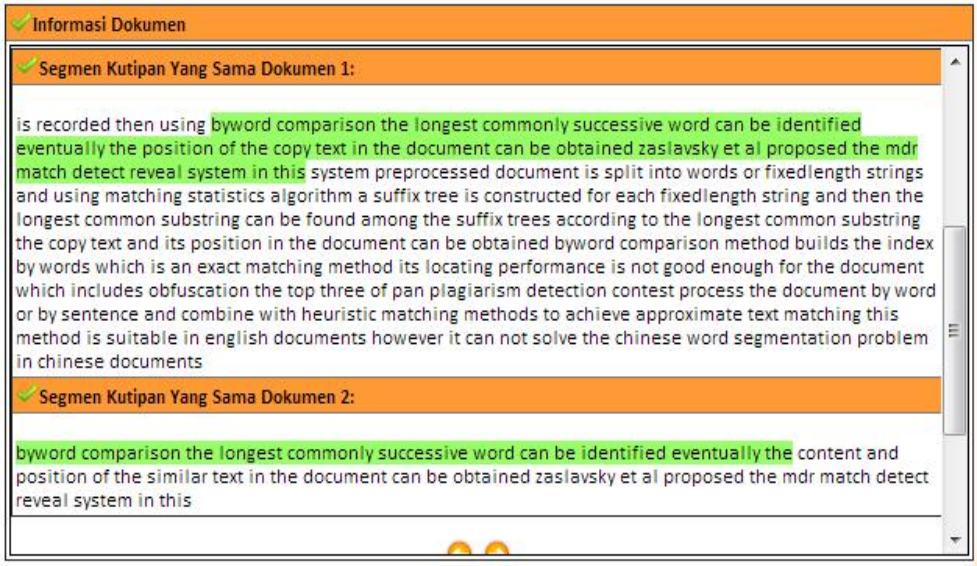
Ada hasil pengujian dari konfigurasi diatas yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.18.



Gambar 5.18 Hasil Kuantitaif Pengujian III

Dari Gambar 5.18 dapat dilihat bahwa dengan menggunakan token berbentuk *biword* dengan batas token terurut >= 6 menghasilkan tingkat kemiripan yaitu 42.31% dengan waktu proses 0.306 detik. Tingkat kemiripan dokumen teks menggunakan konfigurasi ini berkurang dari konfigurasi sebelumnya. Hal ini, karena batas minimal kata yang dianggap sama ditingkatkan menjadi tujuh kata.

Dilihat dari segi kutipan sama yang ditampilkan oleh aplikasi masih terdapat adanya kecocokan token *biword* yang bersifat kebetulan *(coincidental)* walaupun telah meningkatkan batas minimal kata yang dianggap sama. Untuk lebih jelasnya ada kutipan sama di antara dua dokumen teks yang ditampilkan aplikasi dapat dilihat pada Gambar 5.19.



Gambar 5.19 Hasil Kualitatif Pengujian III

Dari beberapa kombinasi yang telah dilakukan menggunakan pendekatan token berbentuk *biword* dengan batas token terurut yang telah dimasukan dapat disimpulkan bahwa pendekatan *biword* kurang baik dalam menemukan kutipan terpanjang yang sama di antara dua dokumen teks walaupun telah meningkatkan batas minimal kata yang dianggap sama. Hal ini, karena token berbentuk *biword* atau dua kata merupakan kata-kata yang masih sering digunakan dalam sebuah dokumen teks sehingga menyebabkan adanya kecocokan *biword* yang bersifat

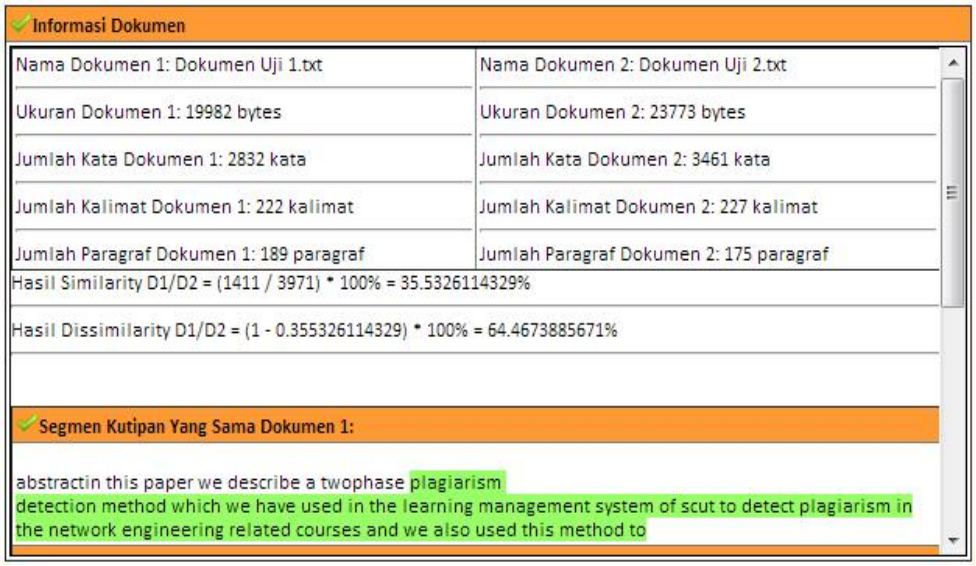
kebetulan *(coincidental)*. Selain itu, pendekatan token berbentuk *biword* juga kurang efektif dalam dalam menemukan pasangan irisan terurut di antara dua dokumen teks.

4. Pengujian IV

Pada pengujian ini digunakan konfigurasi aplikasi pendeteksi plagirisme dokumen teks ini adalah sebagai berikut:

a. Menggunakan token *triword*
b. Menggunakan batas *(threshold)* token terurut >= 3

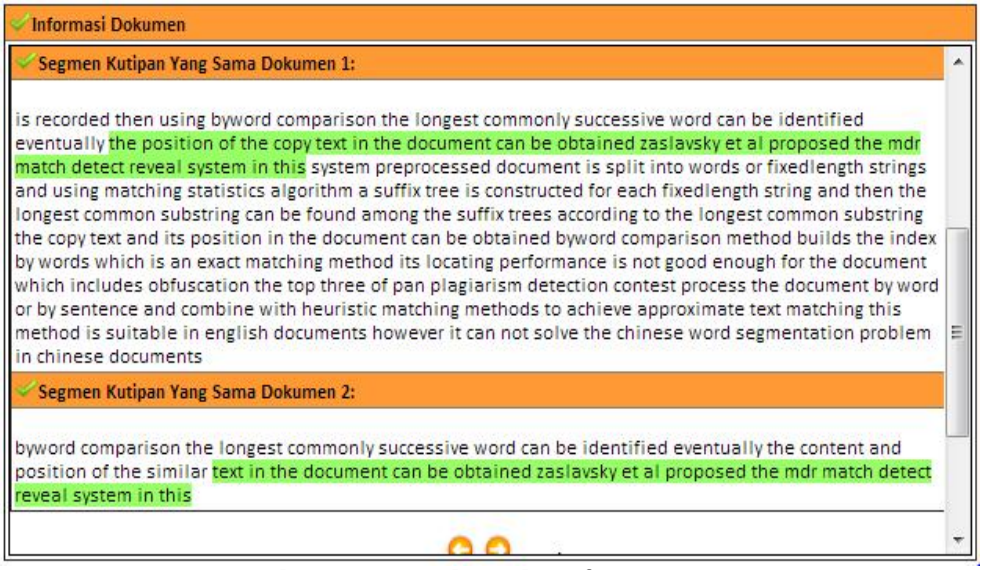
Ada hasil pengujian dari konfigurasi diatas yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.20.



Gambar 5.20 Hasil Kuantitatif Pengujian IV

Dari Gambar 5.20 dapat dilihat bahwa dengan menggunakan token berbentuk *triword* dan batas token terurut >= 3 menghasilkan tingkat kemiripan yaitu 35.53% dengan waktu proses 0.270 detik. Tingkat kemiripan dokumen menggunakan pendekatan *triword* berkurang dibanding dengan menggunakan pendekatan *biword*. Hal ini, karena pencocokan kata dengan menggunakan token berbentuk *triword* memperkecil kemungkinan kecocokan kata yang bersifat kebetulan.

Dilihat dari segi kutipan sama yang ditampilkan oleh aplikasi masih ada terdapat beberapa kecocokan token *triword* yang bersifat kebetulan *(coincidental).* Untuk lebih jelasnya ada kutipan sama di antara dua dokumen teks yang ditampilkan aplikasi dapat dilihat pada Gambar 5.21.



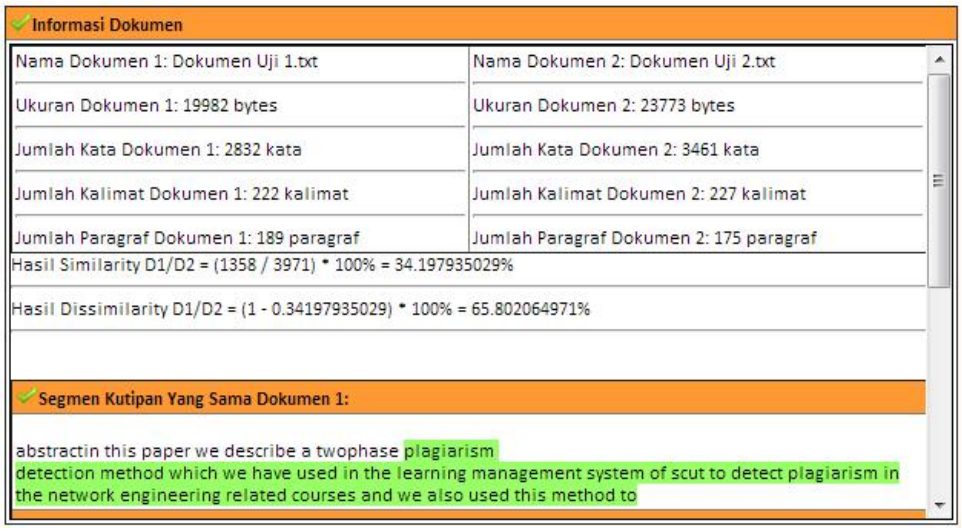
Gambar 5.21 Hasil Kualitatif Pengujian IV

5. Pengujian V

Pada pengujian ini digunakan konfigurasi aplikasi pendeteksi plagirisme dokumen teks ini adalah sebagai berikut:

a. Menggunakan token *triword*
b. Menggunakan batas *(threshold)* token terurut >= 4

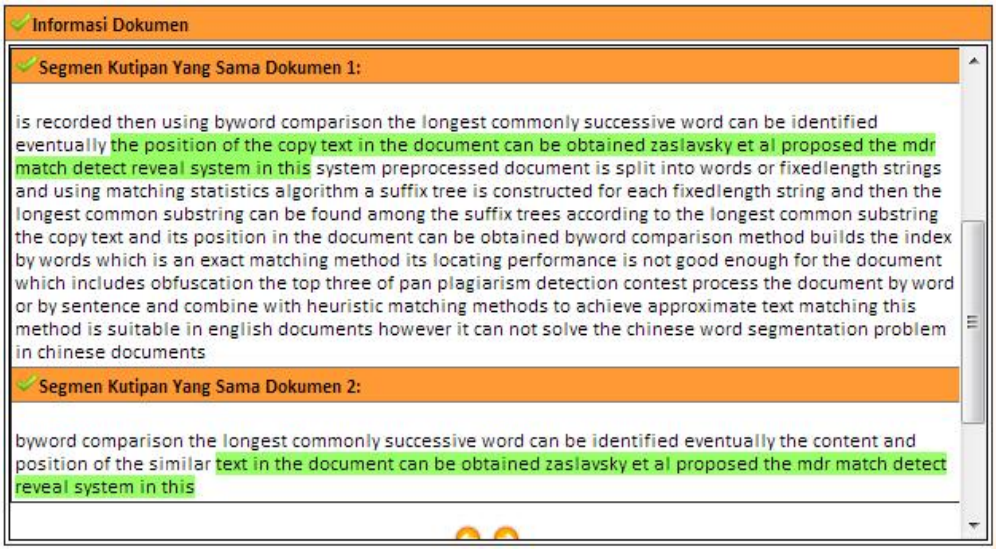
Ada hasil pengujian dari konfigurasi diatas yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.22.



Gambar 5.22 Hasil Kuantitatif Pengujian V

Dari Gambar 5.22 dapat dilihat bahwa dengan menggunakan token berbentuk *triword* dan batas token terurut >= 4 menghasilkan tingkat kemiripan yaitu 34.19% dengan waktu proses 0.303 detik. Berkurangnya tingkat kemiripan dokumen dari konfigurasi sebelumnya karena batas minimal kutipan yang dianggap sama ditingkatkan menjadi enam kata.

Dilihat dari segi kutipan sama yang ditampilkan oleh aplikasi masih ada terdapat beberapa kecocokan token *triword* yang bersifat kebetulan *(coincidental)* walaupun telah meningkatkan batas irisan terurut. Untuk lebih jelasnya, ada kutipan sama di antara dua dokumen teks yang ditampilkan aplikasi dapat dilihat pada Gambar 5.23.



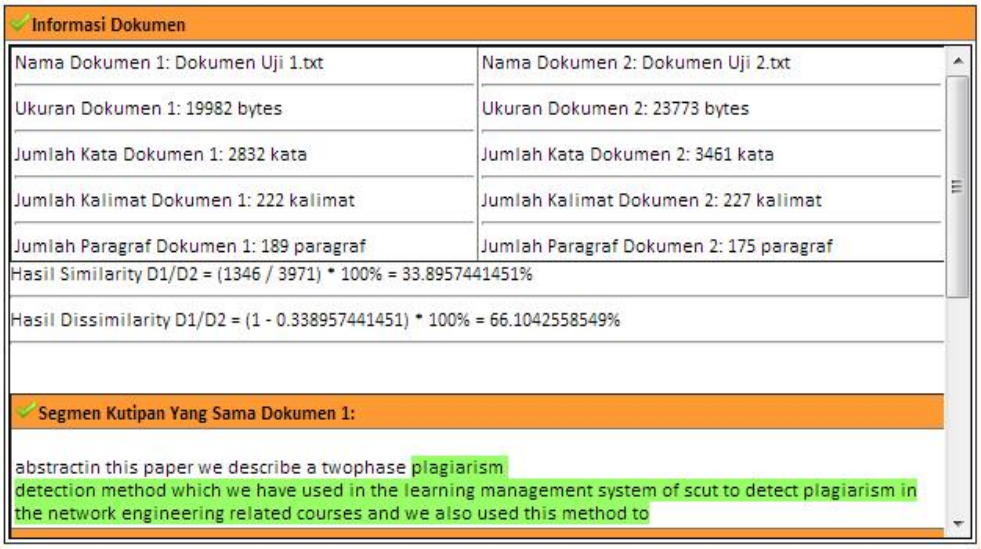
Gambar 5.23 Hasil Kualitatif Pengujian V

6. Pengujian VI

Pada pengujian ini digunakan konfigurasi aplikasi pendeteksi plagirisme dokumen teks ini adalah sebagai berikut:

a. Menggunakan token *triword*
b. Menggunakan batas *(threshold)* token terurut >= 5

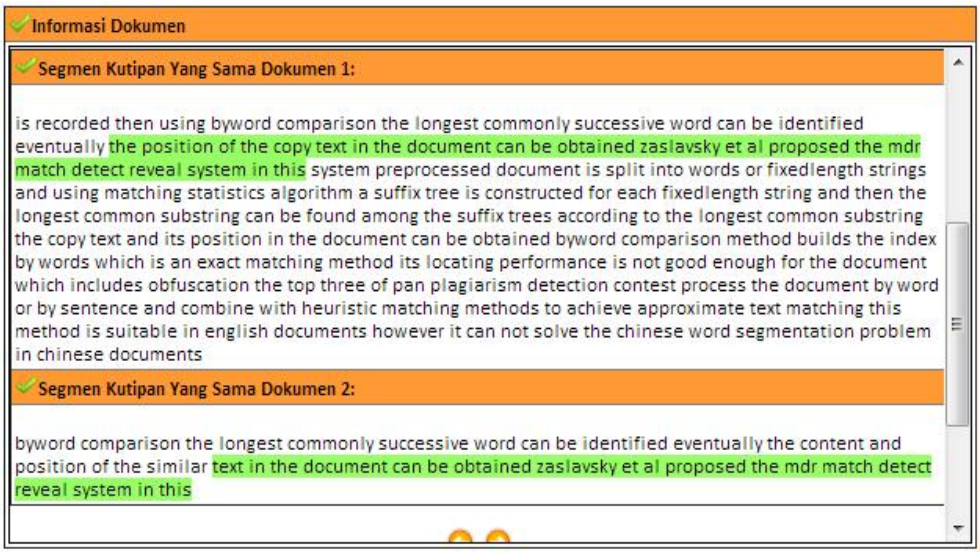
Ada hasil pengujian dari konfigurasi diatas yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.24.



Gambar 5.24 Hasil Kuantitatif Pengujian VI

Dari Gambar 5.24 dapat dilihat bahwa dengan menggunakan token berbentuk *triword* dan batas token terurut >= 5 menghasilkan tingkat kemiripan yaitu 33.89% dengan waktu proses 0.231 detik. Berkurangnya tingkat kemiripan dokumen dari konfigurasi sebelumnya karena batas minimal kutipan yang dianggap sama ditingkatkan menjadi tujuh kata.

Dilihat dari segi kutipan sama yang ditampilkan oleh aplikasi masih ada terdapat beberapa kecocokan token *triword* yang bersifat kebetulan *(coincidental)* walaupun telah meningkatkan batas minimal kutipan yang dianggap sama. Untuk lebih jelasnya, ada kutipan sama di antara dua dokumen teks yang ditampilkan aplikasi dapat dilihat pada Gambar 5.25.



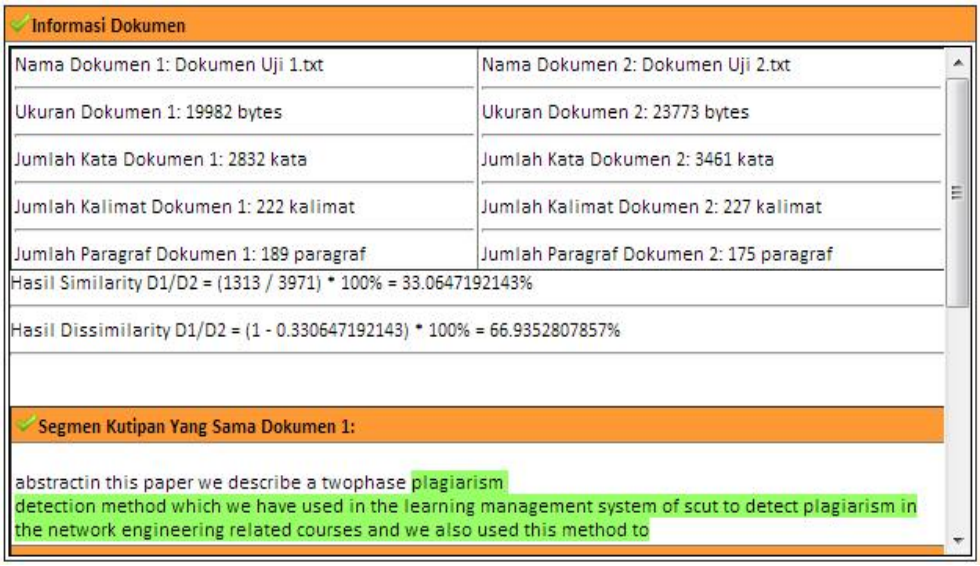
Gambar 5.25 Hasil Kualitatif Pengujian VI

7. Pengujian VII

Pada pengujian ini digunakan konfigurasi aplikasi pendeteksi plagirisme dokumen teks ini adalah sebagai berikut:

a. Menggunakan token *triword*
b. Menggunakan batas *(threshold)* token terurut >= 6

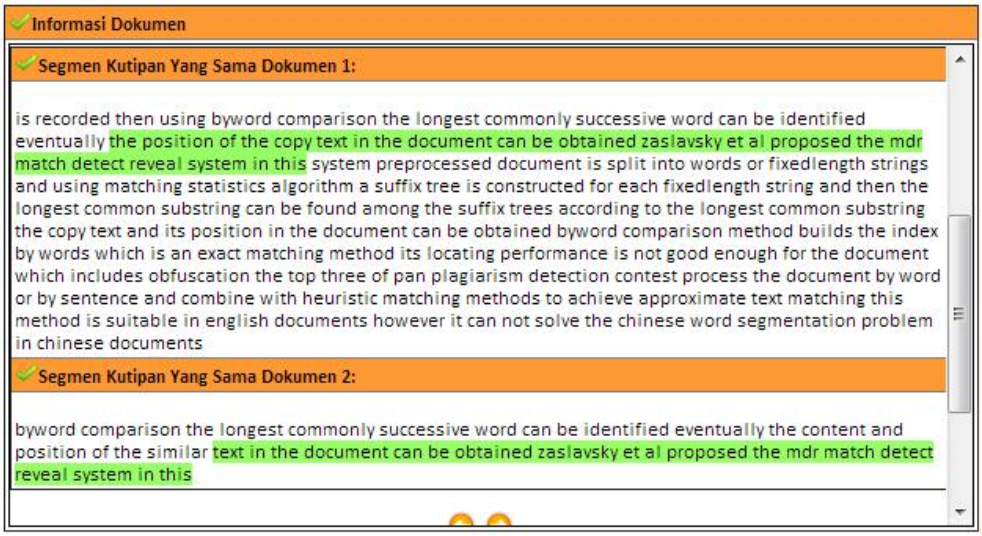
Ada hasil pengujian dari konfigurasi diatas yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.26.



Gambar 5.26 Hasil Kuantitatif Pengujian VII

Dari Gambar 5.26 dapat dilihat bahwa dengan menggunakan token berbentuk *triword* dan batas token terurut >= 6 menghasilkan tingkat kemiripan yaitu 33.06% dengan waktu proses 0,232 detik. Berkurangnya tingkat kemiripan dokumen dari konfigurasi sebelumnya karena batas minimal kutipan yang dianggap sama ditingkatkan menjadi delapan kata.

Dilihat dari segi kutipan sama yang ditampilkan oleh aplikasi masih ada terdapat beberapa kecocokan token *triword* yang bersifat kebetulan *(coincidental).* Untuk lebih jelasnya, ada kutipan sama di antara dua dokumen teks yang ditampilkan aplikasi dapat dilihat pada Gambar 5.27.



Gambar 5.27 Hasil Kualitatif Pengujian VII

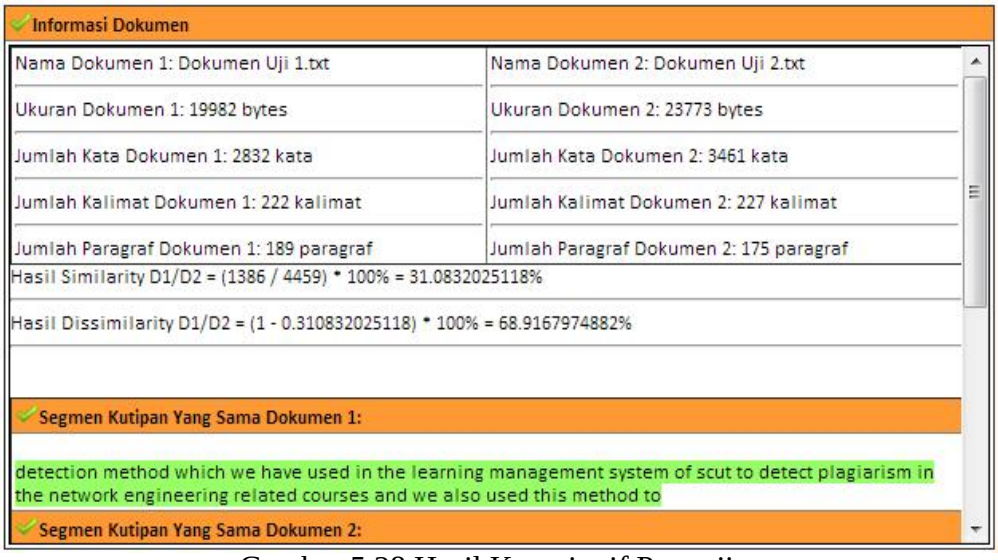
Dari beberapa konfigurasi yang telah dilakukan menggunakan pendekatan token berbentuk *triword* dengan batas token terurut yang telah dimasukan dapat disimpulkan bahwa pendekatan token berbentuk *triword* bekerja lebih baik dibanding pendekatan token berbentuk *biword* dalam menemukan kutipan terpanjang yang dianggap sama di antara dua dokumen teks. Hal ini, karena pendekatan token berbentuk *triword* dapat memperkecil kemungkinan token sama yang bersifat kebetulan *(coincidental)* di antara dua dokumen teks walaupun masih ada beberapa kecocokan token *triword* yang bersifat kebetulan.

8. Pengujian VIII

Pada pengujian ini digunakan konfigurasi aplikasi pendeteksi plagirisme dokumen teks ini adalah sebagai berikut:

a. Menggunakan token *quadword*
b. Menggunakan batas *(threshold)* token terurut >= 2

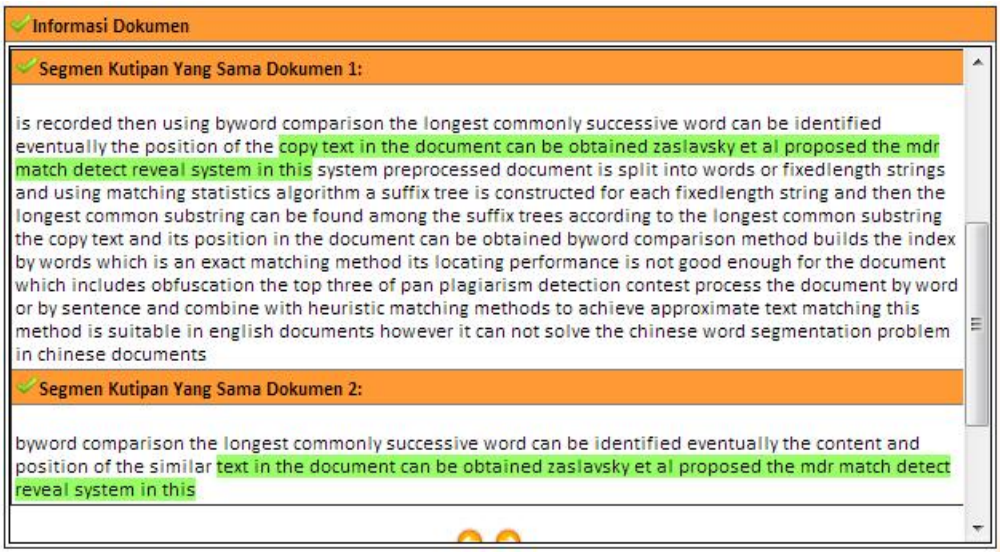
Ada hasil pengujian dari konfigurasi diatas yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.28.



Gambar 5.28 Hasil Kuantitatif Pengujian

Dari Gambar 5.28 dapat dilihat bahwa dengan menggunakan token berbentuk *quadword* dan batas token terurut >= 2 menghasilkan tingkat kemiripan yaitu 31.08% dengan waktu proses 0,301 detik. Berkurangnya tingkat kemiripan dokumen dari pendekatan *triword* disebabkan pencocokan token dilakukan secara empat kata sehingga, lebih memperkecil kemungkinan kecocokan token yang bersifat kebetulan.

Dilihat dari segi kutipan sama yang ditampilkan oleh aplikasi pendekatan token berbentuk *quadword* bekerja lebih baik dari pendekatan token berbentuk *biword* dan *triword.* Untuk lebih jelasnya, ada kutipan sama di antara dua dokumen teks yang ditampilkan aplikasi dapat dilihat pada Gambar 5.29.



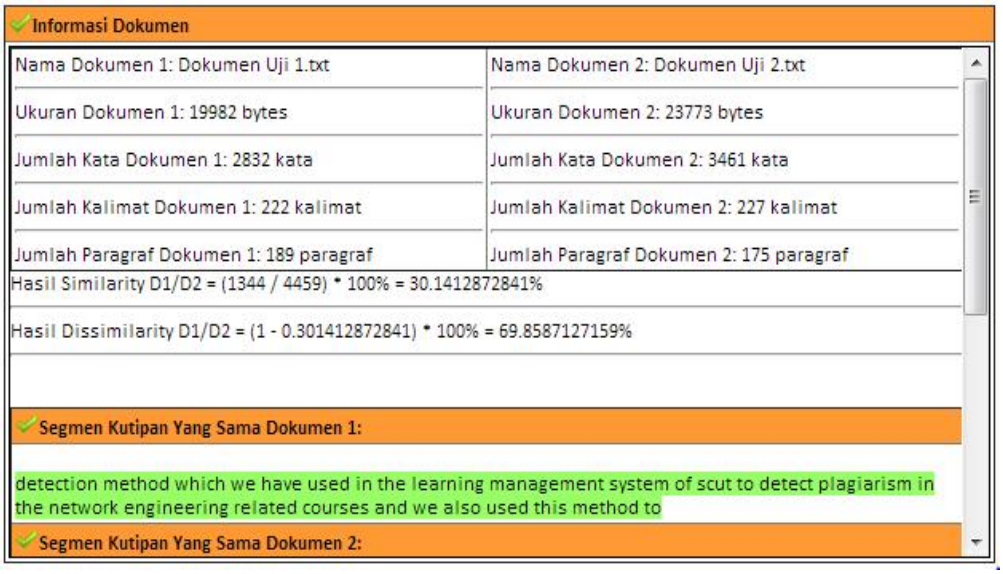
Gambar 2.29 Hasil Kualitatif Pengujian VIII

9. Pengujian IX

Pada pengujian ini digunakan konfigurasi aplikasi pendeteksi plagirisme dokumen teks ini adalah sebagai berikut:

a. Menggunakan token *quadword*
b. Menggunakan batas *(threshold)* token terurut >= 3

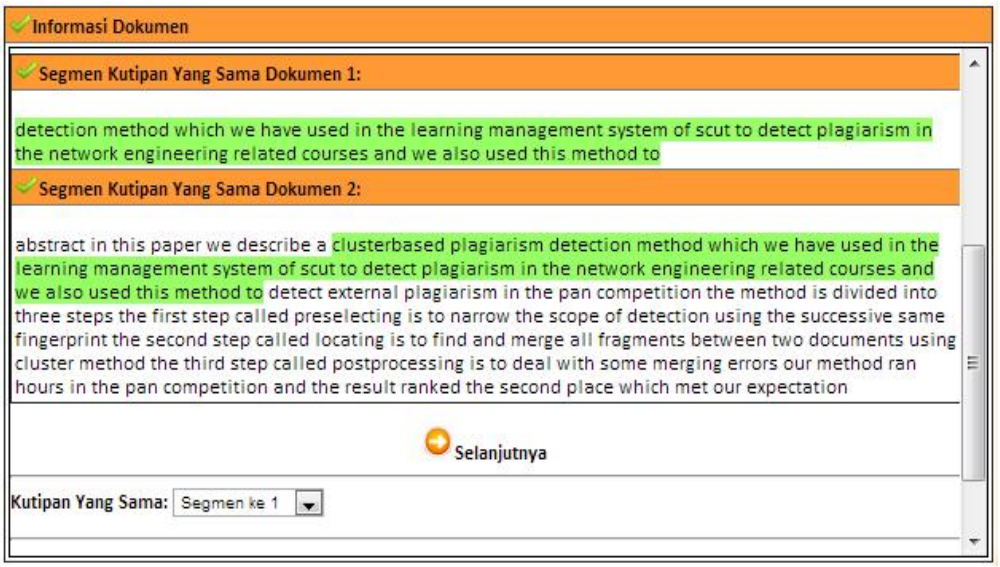
Ada hasil pengujian dari konfigurasi diatas yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.30.



Gambar 5.30 Hasil Kuantitatif Pengujian IX

Dari Gambar 5.30 dapat dilihat bahwa dengan menggunakan token berbentuk *quadword* dan batas token terurut >= 3 menghasilkan tingkat kemiripan yaitu 30.14% dengan waktu proses 0,257 detik. Berkurangnya tingkat kemiripan dokumen dari konfigurasi sebelumnya karena batas kutipan yang dianggap sama ditingkatkan menjadi enam kata.

Untuk lebih jelasnya, ada kutipan sama di antara dua dokumen teks yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.31.



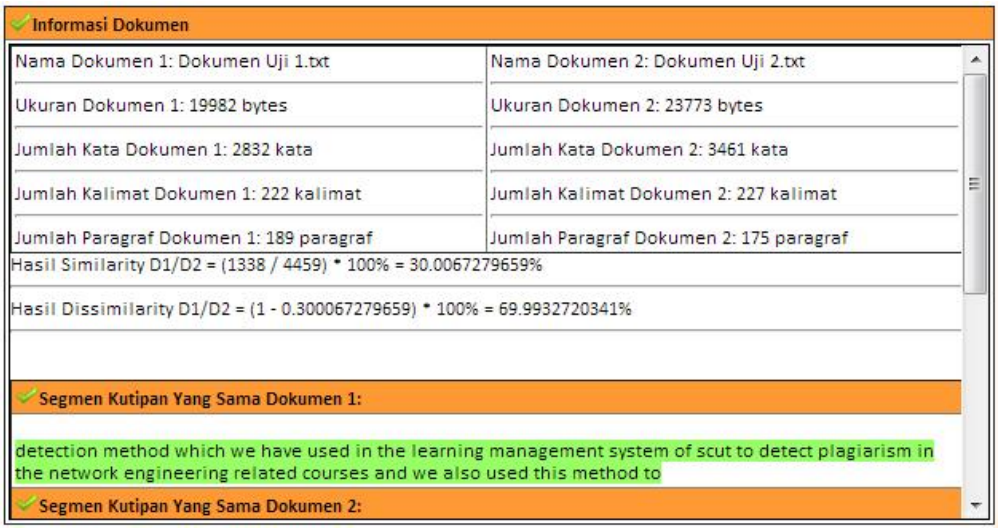
Gambar 5.31 Hasil Kualitatif Pengujian IX

10. Pengujian X

Pada pengujian ini digunakan konfigurasi aplikasi pendeteksi plagirisme dokumen teks ini adalah sebagai berikut:

a. Menggunakan token *quadword*
b. Menggunakan batas *(threshold)* token terurut >= 4

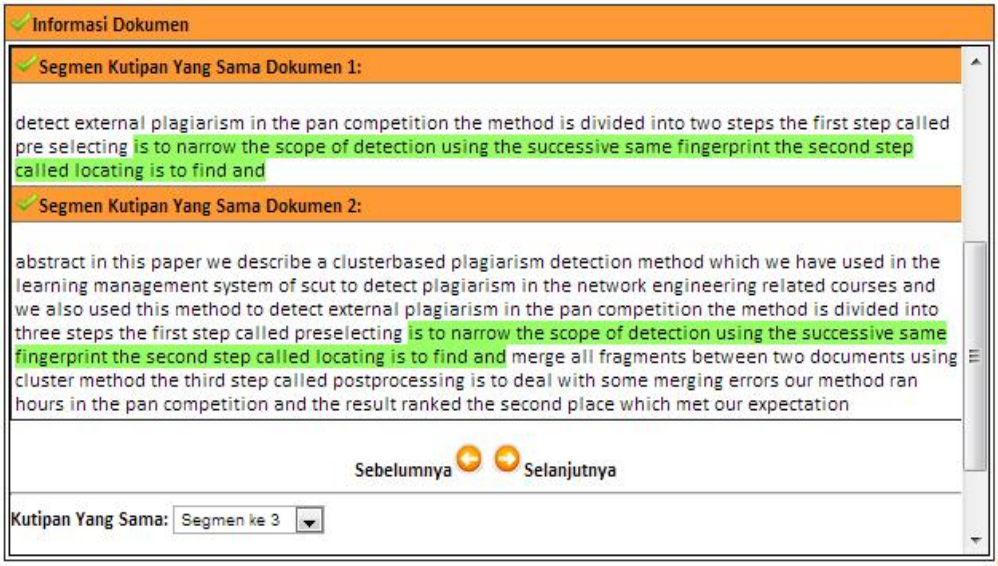
Ada hasil pengujian dari konfigurasi diatas yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.32.



Gambar 5.32 Hasil Kuantitatif Pengujian X

Dari Gambar 5.32 dapat dilihat bahwa dengan menggunakan token berbentuk *quadword* dan batas token terurut >= 4 menghasilkan tingkat kemiripan yaitu 30.00% dengan waktu proses 0,228 detik. Berkurangnya tingkat kemiripan dokumen konfigurasi ini dari konfigurasi sebelumnya karena batas kutipan yang dianggap sama ditingkatkan menjadi tujuh kata.

Untuk lebih jelasnya, ada kutipan sama di antara dua dokumen teks yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.33.



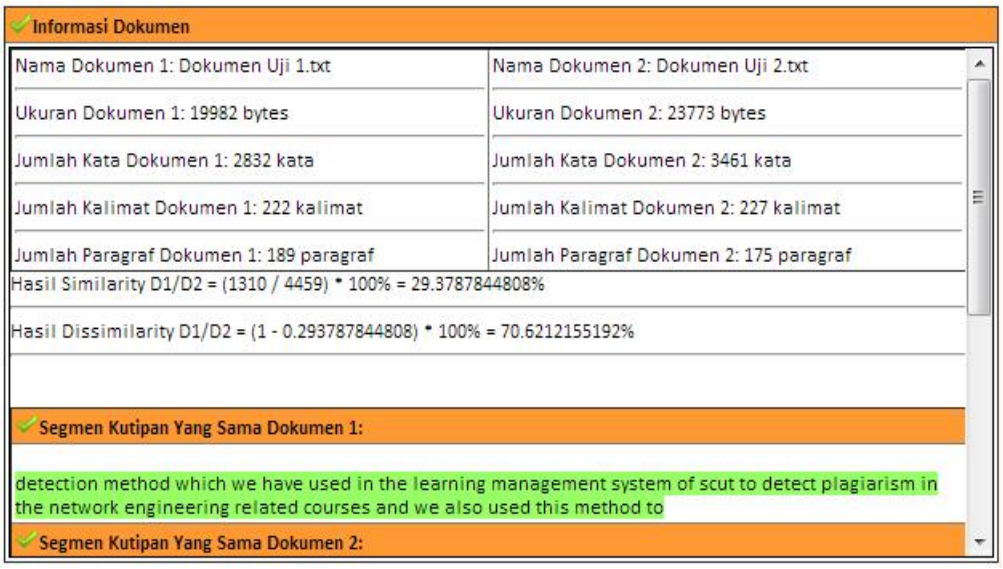
Gambar 5.33 Hasil Kualitatif Pengujian X

11. Pengujian XI

Pada pengujian ini digunakan konfigurasi aplikasi pendeteksi plagirisme dokumen teks ini adalah sebagai berikut:

a. Menggunakan token *quadword*
b. Menggunakan batas *(threshold)* token terurut >= 5

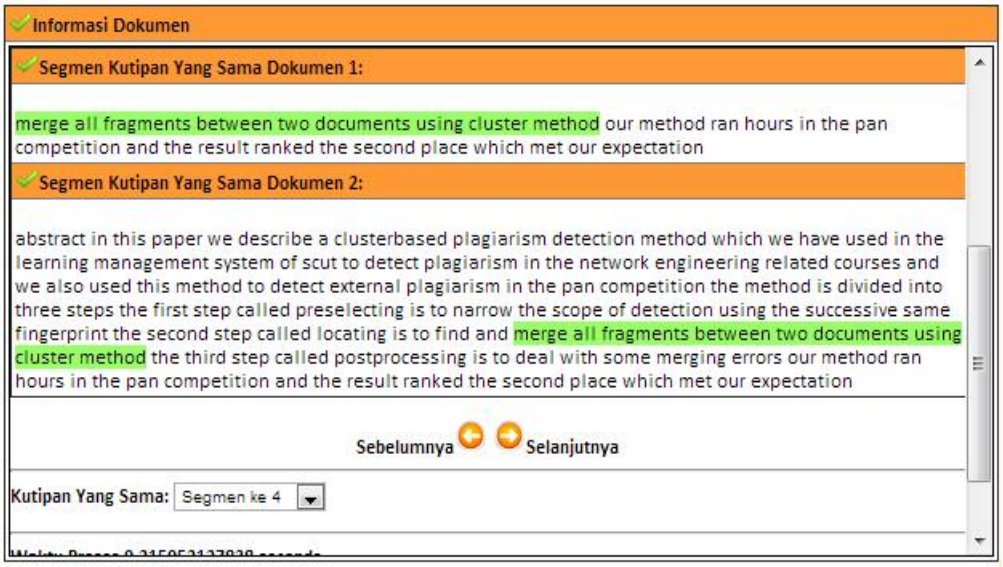
Ada hasil pengujian dari konfigurasi diatas yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.34.



Gambar 5.34 Hasil Kuantitatif Pengujian XI

Dari Gambar 5.34 dapat dilihat bahwa dengan menggunakan token berbentuk *quadword* dan batas token terurut >= 5 menghasilkan tingkat kemiripan yaitu 29.37% dengan waktu proses 0,236 detik. Berkurangnya tingkat kemiripan dokumen dari konfigurasi sebelumnya karena batas kutipan yang dianggap sama ditingkatkan menjadi delapan kata.

Untuk lebih jelasnya, ada kutipan sama di antara dua dokumen teks yang ditampilkan aplikasi dapat dilihat pada Gambar 5.35.



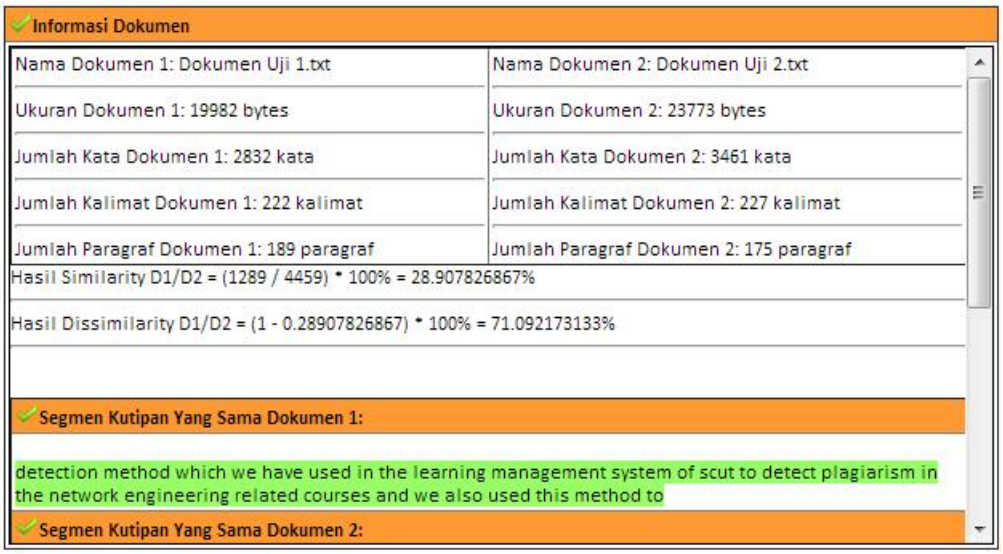
Gambar 5.35 Hasil Kualitatif Pengujian XI

12. Pengujian XII

Pada pengujian ini digunakan konfigurasi aplikasi pendeteksi plagirisme dokumen teks ini adalah sebagai berikut:

a. Menggunakan token *quadword*
b. Menggunakan batas *(threshold)* token terurut >= 6

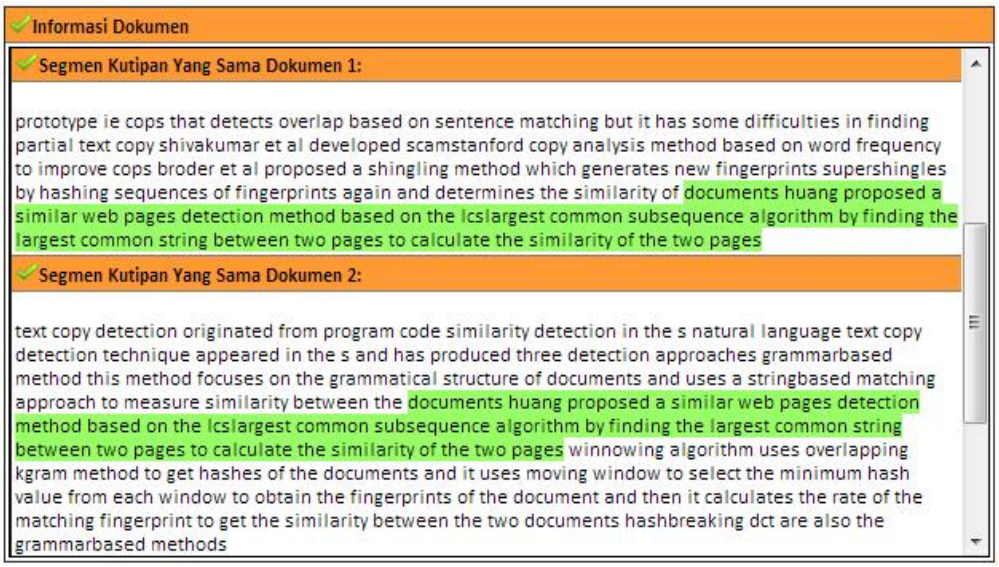
Ada hasil pengujian dari konfigurasi diatas yang ditampilkan oleh aplikasi dapat dilihat pada Gambar 5.36.



Gambar 5.36 Hasil Kuantitatif Pengujian XII

Dari Gambar 5.36 dapat dilihat bahwa dengan menggunakan token berbentuk *quadword* dan batas token terurut >= 6 menghasilkan tingkat kemiripan yaitu 28.90% dengan waktu proses 0,211 detik. Berkurangnya tingkat kemiripan dokumen konfigurasi ini dari konfigurasi sebelumnya karena batas kutipan yang dianggap sama ditingkatkan menjadi sembilan kata.

Untuk lebih jelasnya, ada kutipan sama di antara dua dokumen teks yang ditampilkan aplikasi dapat dilihat pada Gambar 5.37.



Gambar 5.37 Hasil Kualitatif Pengujian XII

Dari beberapa konfigurasi yang telah dilakukan menggunakan pendekatan token berbentuk *quadword* dengan batas token terurut yang telah dimasukan dapat disimpulkan bahwa pendekatan token berbentuk *quadword* bekerja lebih baik dari pendekatan token berbentuk *triword* dan *biword* dalam menemukan kutipan terpanjang yang dianggap sama di antara dua dokumen teks. Hal ini, karena pendekatan token berbentuk *quadword* dapat memperkecil kemungkinan token sama yang bersifat kebetulan *(coincidental)* di antara dua dokumen teks.

5.2.2. Hasil Pegujian

Ada hasil pengujian yang telah dilakukan dengan menggunakan 3 macam dokumen uji dengan beberapa kombinasi yang diujikan adalah sebagai berikut:

1. Menggunakan dokumen uji dua buah ***paper*** **dari penulis Y**. Untuk lebih jelasnya, ada hasil pengujian yang telah dilakukan dapat dilihat pada Tabel 5.6.

Tabel 5.6 Hasil Pengujian Menggunakan ***Paper*** **Penulis Y**

Token Terurut Token	**2**	**3**	**4**	**5**	**6**
	Similaritas (%)				
Biword	-	-	44,53	42,91	42,31
Triword	-	35,53	34,19	33,89	33,06
Quadword	31,08	30,14	30,00	29,37	28,90

2. Menggunakan dokumen uji **BAB II Landasan Teori Laporan Kerja Praktek** dari mahasiswa A dan B. Untuk lebih jelasnya ada hasil pengujian yang telah dilakukan dapat dilihat pada Tabel 5.7.

Tabel 5.7 Hasil Pengujian Menggunakan **BAB II Landasan Teori Laporan Kerja Praktek** Mahasiswa A dan Mahasiswa B

Token Terurut Token	**2**	**3**	**4**	**5**	**6**
	Similaritas (%)				
Biword	-	-	69,40	69,20	69,20
Triword	-	65,75	65,67	65,50	65,50
Quadword	64,47	64,39	64,27	64,27	64,27

3. Menggunakan dokumen uji **BAB I Pendahuluan Laporan Kerja Praktek** dari mahasiswa C dan D dapat dilihat pada Tabel 5.8.

Tabel 5.8 Hasil Pengujian Menggunakan **BAB I Pendahuluan Laporan Kerja Praktek** Mahasiswa C dan Mahasiswa D

Token Terurut / Token	2	3	4	5	6
	Similaritas (%)				
Biword	-	-	12,52	10,50	9,29
Triword	-	10,23	8,65	7,48	7,15
Quadword	8,88	7,65	6.81	6.35	5.58

5.2.3. Kesimpulan Pengujian

Dari beberapa pengujian yang telah dilakukan dengan berbagai kombinasi dapat diambil suatu kesimpulan:

1. Berdasarkan pengujian yang telah dilakukan menggunakan pendekatan token berbentuk *quadword* lebih baik dari pendekatan token berbentuk *triword* dan token berbentuk *biword*. Hal ini, dikarenakan token berbentuk *quadword* dapat memperkecil kecocokan token yang bersifat kebetulan *(coincidental).*
2. Berdasarkan pengujian yang telah dilakukan, mengasumsikan bahwa dengan menggunakan kombinasi pendekatan *quadword* dengan token terurut >=4 sudah cukup untuk mendeteksi kutipan terpanjang yang sama di antara dua dokumen teks, dengan mengamati segi kuantitatif dan segi kualitatif yang dihasilkan dari kombinasi ini.
3. Berdasarkan pengujian yang telah dilakukan, menggunakan pendekatan token berbentuk *quadword* bekerja lebih baik dari pendekatan token berbentuk *triword* dan token berbentuk *biword* dalam menentukan pasangan irisan terurut terutama untuk dokumen yang memiliki tingkat kemiripan tinggi seperti hasil pengujian Tabel 5.7 dengan jumlah kata pada dokumen pertama 2117 kata dan dokumen kedua 2078 kata.
4. Berdasarkan pengujian yang telah dilakukan, penggunaan algoritma *sieve of erasthotenes* cukup baik dalam menemukan pasangan irisan terurut di antara dua dokumen teks walaupun ada beberapa pasangan irisan terurut yang kurang pas dari segi pasangan.

5. Berdasarkan pengujian yang telah dilakukan, aplikasi pendeteksi palgairisme ini dapat mendeteksi plagiarisme *verbatim copy* secara kualitatif dengan melihat langsung di mana letak *copy-paste* yang dilakukan pada paragraf.
6. Berdasarkan pengujian yang telah dilakukan, aplikasi ini dapat dengan tepat menemukan token *biword*, *triword* dan *quadword* yang sama di antara dua dokumen teks dengan mengamati tingkat kemiripan yang dihasilkan dan kutipan terpanjang yang sama ditampilkan oleh aplikasi.
7. Berdasarkan pengujian yang telah dilakukan, semakin besar token yang digunakan dalam mendeteksi plagiarisme dokumen teks maka tingkat kemiripan yang dihasilkan semakin berkurang. Hal ini, dikarenakan pembentukan token *quadword*, lebih sedikit daripada token *triword* dan *biword*. Begitu juga dengan penggunaan batas token terurut, semakin besar batas token terurut yang digunakan maka tingkat kemiripan dokumen teks semakin berkurang.
8. Aplikasi pendeteksi plagiarisme ini hanya memberikan tingkat kemiripan dokumen saja, untuk menentukan atau merekomendasikan dokumen mana yang melakukan plagiat dilakukan secara manual berdasarkan pandangan pengguna salah satunya dengan melihat tanggal pembuatan dokumen.

BAB VI

PENUTUP

6.1 Kesimpulan

Setelah menyelesaikan tahapan-tahapan penelitian aplikasi pendeteksi plagiarisme dokumen teks, dapat diambil beberapa kesimpulan, yaitu:

1. Menggunakan pendekatan token berbentuk *biword*, *triword* dan *quadword* dapat menemukan kutipan terpanjang yang sama di antara dua dokumen teks dan mengukur kemiripan dokumen teks.
2. Menggunakan pendekatan token berbentuk *quadword* lebih baik dari pendekatan token berbentuk *triword* dan token berbentuk *biword*. Hal ini, dikarenakan token berbentuk *quadword* dapat memperkecil kecocokan token yang bersifat kebetulan *(coincidental).*
3. Berdasarkan penelitian yang telah dilakukan, mengasumsikan bahwa dengan menggunakan kombinasi pendekatan *quadword* dengan token terurut >=4 sudah cukup untuk mendeteksi kutipan terpanjang yang sama di antara dua dokumen teks, dengan mengamati segi kuantitatif dan segi kualitatif yang dihasilkan dari kombinasi ini.
4. Berdasarkan penelitian yang telah dilakukan, penggunaan algoritma *sieve of erastothenes* cukup baik dalam menemukan dan mempercepat pencarian pasangan irisan terurut di antara dua dokumen teks.
5. Menggunakan pendekatan token berbentuk *quadword* lebih baik dari pendekatan token berbentuk *triword* dan token berbentuk *biword* dalam menemukan pasangan irisan terurut terutama untuk dokumen yang memiliki tingkat kemiripan tinggi.
6. Berdasarkan penelitian yang telah dilakukan, pendekatan token berbentuk *biword, triword* dan *quadword* memenuhi kebutuhan mendasar algoritma pendeteksi plagiarisme dokumen teks yaitu *whitespace insensitivity, noise surpression* dan *position independence.*

6.2 Saran

Berdasarkan penelitian yang telah dilakukan, adapun saran-saran yang dapat dilakukan untuk perbaikan dan pengembangan aplikasi pendeteksi plagiarisme dokumen teks, yaitu:

1. Aplikasi ini dapat dikembangkan dengan menambahkan pendeteksian *sinonim* karena tindak plagirisme sering kali dilakukan dengan menggunakan persamaan kata.
2. Aplikasi ini masih bersifat *one to one* dalam mendeteksi plagiarisme dokumen sehingga kurang efisien jika dokumen yang akan dideteksi berjumlah banyak. Kedepannya, aplikasi yang dibangun dapat mendeteksi kemiripan dokumen secara *many to many*.
3. Aplikasi yang dibangun kedepannya dapat mendeteksi kalimat aktif dan kalimat pasif karena tindak plagirisme sering kali dilakukan dengan mengubah kalimat aktif menjadi pasif atau sebaliknya.

DAFTAR PUSTAKA

Elbegbayan, Norzima, *Winnowing, A Document Fingerprint Algorithm*. Department of Computer Science, Linkoping University. 2005 [Available]: online http://www.ida.liu.se/~TDDC03/oldprojects/2005/final-projects/prj10.pdf , diakses tanggal 26 Mei 2012

Himawan, Hidayatulah. *Aplikasi Teknologi Green Computing Melalui Optimalisasi Kinerja Komputer*. Yogyakarta: Jurusan Teknik Informatika UPN "Veteran" Yogyakarta. 2010 [Available]: online http://repository.upnyk.ac.id/1972/1/4_IWAN_optimalisasi_kinerja_komputer_menggunakan_processor_tunggal.pdf, diakses tanggal 1 September 2012

Kent, Chow Kok and Naomie Salim. *Features Based Text Similarity Detection*. Malaysia: *Faculty of Computer Science and Information Systems, University Teknologi Malaysia*. 2010 [Available]: online http://arxiv.org/ftp/arxiv/papers/1001/1001.3487.pdf, diakses tanggal 26 Juli 2012

Kusmawan, Putu Yuwono, Umi Laili Yuhana dan Diana Purwitasari. *Aplikasi Pendeteksi Penjiplakan Pada File Teks Dengan Algoritma Winnowing*. Surabaya: Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh November Surabaya. 2010 [Available]: online http://digilib.its.ac.id/public/ITS-Undergraduate-10278-Paper.pdf , diakses tanggal 17 Mei 2012

Kurniawati, Ana, Wicaksana, I Wayan Simri. *Perbandingan Pendekatan Deteksi Plagiarism Dokumen Dalam Bahasa Inggris*. Jakarta: Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Gunadarma. 2008 [Available]: online http://research.mercubuana.ac.id/proceeding/OSSOC_26.pdf, diakses tanggal 25 Mei 2012

Lyon C, Malcolm J, and Dickerson B. *Detecting short passages of similar text in large document collections*. Hertfordshire: Proceedings of EMNLP (Empirical Methods in Natural Language Processing). 2001 [Available]: online https://uhra.herts.ac.uk/dspace/bitstream/2299/1695/1/901890.pdf, diakses tanggal 29 Mei 2012

Manning, Christopher D., Prabhakar Raghavan dan Hinrich Schütze. *An Introduction to Information Retrieval*. England: Cambridge University Press. 2009

Martin, B. *Plagiarism: a misplaced emphasis, Journal of Information Ethics*. 1994

Munir, Rinaldi. *Algoritma Dan Pemograman*. Bandung: Informatika Bandung. 2007

Nugroho, Eko. *Perancangan Sistem Deteksi Plagiarisme Dokumen Teks Dengan Menggunakan Algoritma Rabin-Karp*. Malang: Program Studi Ilmu Komputer, Jurusan Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Brawijaya. 2011
[Available]: online
http://blog.ub.ac.id/ecoorner/files/2011/03/Bab12345.pdf, diakses tanggal 17 Mei 2012

Purwitasari, Diana, Putu Yuwono Kusmawan, Umi Laili Yuhana. *Deteksi Keberadaan Kalimat Sama Sebagai Indikasi Penjiplakan Dengan Algoritma Hashing Berbasis N-gram*. Surabaya: Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh November Surabaya. 2011
[Available]: online
http://kursor.trunojoyo.ac.id/wp-content/uploads/2012/03/vol6_no1_p5.pdf diakses tanggal 17 Mei 2012

Richard M. Karp and Michael O. Rabin. *Eficeient Randomized Pattern-matching algorithms*. USA: IBM Journal of Research and Development. 1987
[Available]: online
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.86.9502&rep=rep1&type=pdf, diakses tanggal 28 September 2012

Schleimer, Saul, Daniel S. Wilkerson, and Alex Aiken. *Winnowing: Local Algorithms for Document Fingerprinting.* San Diego: In Proceedings of the ACM SIGMOD International Conference On Management Of Data. 2003
[Available]: online
http://theory.stanford.edu/~aiken/publications/papers/sigmod03.pdf,
diakses tanggal 17 Mei 2012

Sediyono, Agung dan KRK Mahamud. *Algorithm of the Longest Commonly Consecutive Word for Plagiarism Detection in Text Based Document*. Digital Information Management. 2008

Udi Manber. Finding similar files in a large file system. San Fransisco: In *Proceedings of the USENIX Winter*. 1994

[Available]: online
http://webglimpse.net/pubs/TR93-33.pdf, diakses tanggal 28 September 2012

Wang Tao, Fan Xiao-Zhong, Liu Jie, *Plagiarism Detection in Chinese Based on Chunk and Paragraph Weight*. Kunming: in Proceedings of the Seventh International Conference on Machine Learning and Cybernetics. 2008

Zou, Du, Wei-jiang Long and Zhang Ling. *A Cluster-Based Plagiarism Detection Method.* China: Lab Report for PAN at CLEF. 2010
[Available]: online
http://clef2010.org/resources/proceedings/clef2010labs_submission_7.pdf, diakses tanggal 30 Mei 2012